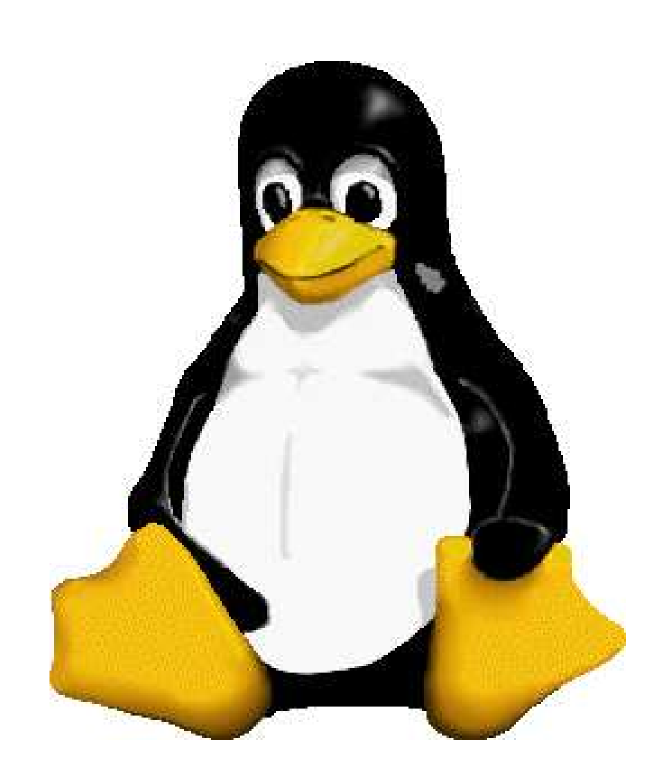
# Instalación y Administración de Sistemas



## Linux

Georg Lehner

19/08/2000

# Índice general

# 1. Prefacio

Este manual se ha percebido como acompañante de los cursos sobre Linux que el autor ha dado en diferentes instituciones. Como tal no corresponde a un tema específico de computación, de Linux o de Hardware, sino se comprende como referencia rápida para problemas de uso comunes en torno al uso diario de una computadora, en este caso compatibles PC con el sistema GNU/Linux. Pretendo proveer los conocimientos necesarios para poder manejar, configurar e instalar un sistema Gnu/Linux a personas usuarias y administradores de computadoras en redes TCP/IP y/o MS-Windows, que usan, o planifican usar su computadoras en ámbitos profesionales. Para este fin el manual es diseñado también como introducción a los sistemas Unix y X-windows para facilitar la comprensión a quienes solamente conocen sistemas operativos diferentes. El contenido abarca temas como: manejo básico de un sistema Unix desde el shell, programación shell básica, manejo básico del sistema X-Windows, configuración de hardware, instalación y optimización del sistema, manejo de cuentas de usuarios, configuración y mantenimiento de redes TCP/IP y MS-Net bajo Linux, configuración de Modem y cuentas PPP, configuración del sistema de impresión remoto lpr, Ghostscript y utilidades.

Se requieren conocimientos avanzados de usuario de computadoras en ambiente texto y gráfico (Windows), y conocimiento de Inglés técnico en lectura.

La información presentada no es ni completa ni necesariamente es aplicable o correcta para cualquier sistema Linux, además del hecho, que GNU/Linux y sus distribuciones son un "blanco móvil", es decir presentan una evolución rápida y lo que hoy puede ser la solución estándard para un problema de administración de computadoras, mañana ha pasado a ser obsoleto – y así los manuales que describen estas soluciones.

Aprovecho este lugar para manifestar, que recibo con gusto sugerencias y correcciones, los cuales pueden dirigirse a mi dirección electróncia: `Jorge.Lehner@gmx.net`. Puede encontrar una versión html de este documento en Internet en la siguiente dirección: `http://www.magma.com/linuxbook/`.

Quiero agradecer a dos personas, que no conozco personalmente, pero los considero muy especiales y que han hecho posible y necesario este manual, a través de crear un sistema operativo - Linux, y una nueva forma de considerar como compartir la propiedad intelectual y protegerla a la vez - Software Libre. Estas personas son Linus Torvalds, que no se contentó con los sistemas operativos ya existentes y tomo el valor de crear uno de nuevo, y Richard Stallman, que no se contento con las alternativas existentes - comprar Software o robarlo, y creó una alternativa, la de compartirlo. Estos dos nombres solo son símbolos para miles más, que contribuyeron a hacer realidad estas dos visiones y mi principal agradecimiento es por vivir en el presente y presenciar la maravilla, que en un nuestro mundo considerado meramente egoista se manifiesta esta alternativa.

# 2. Introducción

El sistema operativo GNU/Linux consiste de dos partes: el núcleo Linux, creado inicialmente por Linux Torvalds como un proyecto de un aficionado, y las aplicaciones del proyecto GNU, que complementan el núcleo para formar un sistema operativo completo.

En conjunto con la percepción de la licencia libre GPL y sus parecidos Linux ha convertido en un fenómeno. Dentro de un tiempo muy corto ha evolucionado, para convertirse en una verdadera alternativa a las soluciones comerciales en el sector de sistemas operativos profesionales.

Este manual está dividido en tres partes. en la primera parte se encuentra una introducción básica para el manejo de un sistema Unix[1] en ambiente texto y en ambiente gráfico. Además cuenta con un pequeño instructivo sobre la programación del interpretador de comandos bash.

La segunda parte está dedicada a la configuración de los diferentes componentes de un sistema Unix/Linux.

Primero se revisa con detalle la configuración del Servidor X[2] y las diferentes formas de su empleo y después la configuración de los servicios básicos: redes, conexión telefónica al Internet, impresión. El siguiente tema da una visión sobre el uso y la configuración de diferentes protocolos de redes ajenos a - pero disponibles en Linux y por último se tocan temas relacionados con la configuración de cuentas de usuarios.

La tercera parte se dedica a la instalación del sistema Linux, y a la configuración de Hardware específico y no soportado por la instalación básica. Así como la agregación y la deconfiguración de programas y aplicaciones.

En el anexo se encuentran materiales de referencia, que no solo apoyan la lectura del texto, sino sirven de guia para el manejo del sistema: Referencia rápida de los editores Emacs y vi, FHS, un glosario de términos y otra información (hojala) útil.

[1] La mayoría de los temas no son específico para Linux, sino aplican a cualquier sistema compatible con POSIX (vea también 150).

[2] El "driver" del ambiente grafico de Linux, parte del X-Window-System

# 3. Convenciones usadas en el libro

english terms son términos que preferiblemente se memorizan en idioma inglés, por su común uso. A veces solo se introducen una vez en español, y despúes se siguen usando en el texto.

`commandos` y `nombres de archivos` se escribe en letra `Courier` (typewriter). En realidad, todo lo que puede aparecer en una pantalla de computadora, o impreso, a excepción de:

*variables* y *parametros*. Estos se escriben en letra *cursiva*. De esta manera nos referimos a texto, que tiene que ser ingresado según los requerimientos del caso.

$ *comandos...* simboliza una sesión de interpretador de comando (shell) interactivo, como usuario normal – notese el símbolo $.

# *comandos...* simboliza una sesión como usuario administrativo, superuser, o en corto: root. Los comandos indicados deben de ejecutarse con los privilegios del superuser – notese el símbolo #.

¿Tiempo justo para un ejemplo?

```
$ cp origen destino
```

significa, ejecutar como usuario no privilegiado el comando `cp`, dándole dos parámetros *origen* y *destino*. En un dado caso, *origen* podría ser el archivo `carta.txt`, y *destino* otro archivo `carta_nueva.txt`.

`[Enter]` y parecidos, simbolízan las téclas respectivas de un teclado de computadora. Otros ejemplos: `[1]` el número uno, `[a]` la letra menuscula a, `[Ins]` la tecla para insertar un espacio.

La notación BNF es muy común y práctica para describir cómo ingresar un comando con todos sus parametros obligatorios y opcionales y la usarémos cuando sea necesario. Puede verse mas detalle en el anexo de este libro (F).

# Parte I.

# Manejo

# 4. Manejo de Linux en ambiente textual

## 4.1. login

En un sistema multiusuario (como Linux) tienen que diferenciarse los espacios (ambientes de trabajo) de los diferentes usuarios. Esto es una función del login que se encuentra como portal cada vez que se accede a la computadora. El segundo objetivo del login es la autentificación del usuario, o sea, detectar a través de una clave (secreta)=Password que la persona que utiliza el espacio de usuario indicado es este mismo. Es una medida básica de protección de los datos individuales de cada usuario.

Posibles formas de presentación son:

```
login: nombre
Password:
```

o también

```
Username: nombre
Password:
```

El nombre del usuario es visible, pero la clave no se repite en la pantalla. Si, y solamente si, el par *nombre*/*clave* coincide con un par de valores grabados en el sistema se provee acceso. En cualquier otro caso se denega el acceso. Es otra medida de seguridad que no se indica si fallo el *nombre* o la *clave*.

El *nombre* designa una cuenta (account) que es el espacio o ambiente de trabajo del usuario.

## 4.2. shell

El sistema prove directamente después de la autentificación una facilidad para interactuar, es decir efectuar comandos y visualizar las respuestas correspondientes del sistema. Esta facilidad se llama procesador de linea de commandos - command line processor -, o "shell" en la nomenclatura de unix. "shell" signigica la cáscara de la concha y es un sinónimo para la capa visible del sistema operativo.

El shell lee una línea de texto que el usuario teclea. Al pulsar la tecla `[Enter]` interpreta el texto (por lo que también se le llama 'interpretador de línea de commandos=command line interpreter) y ejecuta los comandos encontrados. Si se cometió un error lo indica así. Al terminar los comandos el shell muestra una cadena de identificación llamado "prompt" y procede a leer la siguiente línea.

Existen diferentes shell, los tradicionales se llaman 'sh' y 'csh', otros son:bash, ash, zsh, ksh, csh, tcsh,... nosotros trabajamos unicamente con bash, que es compatible con sh y el shell official del software GNU.

## 4.3. línea de commandos

En una línea de commando se escriben todos los elementos de un comando, separados por espacio en blanco (`[Espacio]`). Pueden concatenarse varios commandos, separados por el símbolo '`;`' (semicolon) en una sola línea.

Un commando tiene tres partes: *nombre*, *opciones* y *parámetros*. Solamente el nombre es obligatorio. Ejemplo:

```
ls     -l --all   *.c
nombre [opciones] [parámetros]
```

**nombre** nombre del comando. Puede ser el nombre de cualquier archivo de programa. Ejemplo: '`ls`'

**opciones** indicacan al comando, que tiene que modificar su comportamiento de una cierta forma. La forma tradicional de introducir una opción es una sola letra, introducida con el símbolo '-' (guión). Ejemplo: `-l`

El proyecto GNU ha introducido opciones explicativas (long options) que consisten de una palabra entera y estan introducidas por dos guiones consecutivos: '`--`'. Ejemplo: `--all`

Opciones por su parte también pueden tener parámetros.

Ejemplos: -`s15`, `--with_gettext=yes`

**parámetros** indican los objetos sobre cuales el comando debe actuar. Normalmente nombres de archivos o de directorios. Ejemplo: '`*.c`'

Tradicionalmente tienen que escribirse los parámetros antes de los parámetros. En los programas GNU muchas veces eso no está exigido, pero es buena práctica hacerlo de todas formas.

**Wildcards o comodínes** son símbolos que pueden aparecer en los nombres de archivos, a veces también en los de los directorios. El comodín es sustituido por el shell con todos los nombres de archivos que cumplen con su forma.

***** sustituye una cadena arbitraria de caracteres.

**?** sustituye cualquier caracter individual.

En el ejemplo de arriba se sustituye '`*.c`' con todos los archivos del directorio actual que terminan en los caracteres punto - '.' y 'c'.

## 4.4. comandos básicos

### Manipulación de directorios

| | |
|---|---|
| ls | listado de archivo en un directorio |
| rmdir *nombre* | remover un directorio (vacio) |
| mkdir *nombre* | crear un directorio |
| cd *nombre* | "entrar" en un directorio; cambiar el directorio actual al indicado |
| find | listar (y manipular) seleccionados archivos en un subdirectorio |
| locate | encontrar archivos en todo el (los) disco(s) |
| du | (diskusage) calcular el espacio consumido por todos los archivos de un determinado subdirectorio |
| df | (disk free) calcular el espacio libre en un disco duro |

### Manipulación de archivos

| | |
|---|---|
| rm | remover uno o varios archivos |
| cp | copiar uno o varios archivos y/o directorios |
| mv | mover uno o varios archivos y/o directorios, o renombrar un archivo o directorio |
| cat | editor y visualizador primitivo |
| grep | buscar textos en archivos |

### Obtener ayuda

*comando* `--help` muestra una descripción corta del comando

man *comando* muestra la página del manual de unix que describe el commando

info *comando* visualizar ayuda info o man sobre el comando

info por si mismo contiene un manual extensivo con hiperenlaces, ejemplos y mucho más para todos los programas GNU

## 4.5. Manipulación de archivos

El objeto principal en el uso de los sistemas operativos son los archivos. Los comandos mas usados para manipularlos se discuten aqui en brevedad.

### 4.5.1. cp

'`cp`' puede copiar una **lista de archivos**, explícitamente nombrada, como directorio, o con wildcards hacia otro **directorio**. Si el directorio no existe se crea. No se pueden especificar jerarquías de directorios como destino, el comando

```
$ cp documentos/carta.txt archivo/2003/Marzo/
```

falla si `archivo`, `archivo/2003` y `archivo/2003/Marzo` no existen. Por otro lado el comando

```
$ cp documentos/carta.txt archivo/2003/Marzo
```

es exitoso si solo el directorio `archivo/2003` existe, creando una copia de `carta.txt` con el nombre `Marzo`. Se puede copiar un archivo a otro, que tiene un nombre diferente. Si el destino existe se sobreescribe (!).

### 4.5.2. ln

'`ln`' crea enlaces (links). Esto no son archivos, sino referencias hacia archivos, que se pueden imaginar como entradas duplificadas en un directorio. Existen enlaces simbólicos (soft link) y directos (hard link). Los primeros son como hiperenlaces en la web: un cadena de caracteres que refleja la ubicación original de un archivo o directorio, los segundos son copias directas. Por lo general se prefieren usar soft links.

```
ln -s
```

Una vez que llega a tener práctica descubrira que el comando `mv` es utilizado mas veces que el comando `cp`. `mv` tiene la misma funciónalidad de ubicar un archivo en otro lado, mas se elimina el archivo original. `mv` es mucho más eficiente que `cp`, ya que solo tiene que cambiar una entrada en el directorio, y no necesita hacer copia del contenido del (posiblemente largo) archivo.

### 4.5.3. find

`find` itera por todos los archivos del directorio especificado, o por omisión por el actual, y repite esta función recursivamente por todos los subdirectorios. Si no se especifica una operación, `find` imprime los nombres de archivos y directorios encontrados. Este comportamiento se puede modificar, usando operadores en forma de opciones con parámetros. Ejemplos:

`find` genera una lista de todos los archivos en el directorio actual y en sus subdirectorios

`find /home -name core -exec rm '{}' \;` encuentra todos los archivos de nombre `core` bajo el directorio `/home` y ejecuta el comando `rm` con la vía de acceso del archivo `core` correspondiente como parametro, es decir borra todos los archivos `core`. La secuencia `'{}'` denota el nombre del archivo actualmente encontrado y la secuencia `\;` es necesaria para concluir el parametro de `-exec`.

`find /home -name core -atime 7 -print -exec -a rm '{}' \;` tests secuenciales se combinan implícitamente con una operación 'y' (AND). En este ejemplo, todos los archivos `core` en todos los subdirectorios de `/home` con una edad de mayor de siete días se eliminan. `-print`: ademas imprime el nombre del archivo en la pantalla.

`find` es un comando muy poderoso y se utiliza mucho en la administración automatizada de sistemas. Para encontrar la ubicación de un cierto archivo, por otro lado, es muy ineficiente ya que itera cada vez por todo el disco duro. Para encontrar la ubicación de un archivo se utiliza mejor el comando `locate`. Este comando utiliza una base de datos indexados de todos los archivos en el disco duro, y extrae de esta la ubicación. Ejemplo:

`locate fernando` este comando imprime la via de acceso completa de todos los archivos que en alguna parte de su nombre contienen la cadena de caracteres 'fernando'.

Cuando un archivo es renombrado, borrado, se crea un nuevo archivo, etc. la base de datos no se actualiza automáticamente. Por eso `locate` no reporta correctamente las ubicaciones, después de un cierto tiempo de uso. El programa `updatedb` que se ejecuta automáticamente una vez por semana actualiza esta base de datos. `locate` utiliza `find` para registrar todos los archivos que se encuentran en el (los) disco(s).

### 4.5.4. echo y cat

Consta que los comandos básicos que hemos visto hasta ahora mas que todo tratan con la destrucción de archivos o directorios: `rm`, `rmdir` etc. y que no hemos visto comandos para **crear** archivos. Una razón es, que para la creación arbitraria de archivos a mano se utilizan programas dedicados, según el tipo de datos a crear: programas de dibujo gráfico, captura de imágenes, sonido, etc. o en el caso de programas o textos sencillos: editores.

Por otro lado se pueden crear pequeños archivos mediante comandos sencillos, haciendo uso de las funciones de redireccionamiento del shell:

`echo` *texto arbitrario* imprime el *texto arbitrario* en la pantalla. Eso parece trivial pero no lo es si se emplea en un contexto respectivo. Dos ejemplos son, la impresión de mensajes de aviso en scripts, lo que se explicará a fondo en el capítulo 6, y la creación de pequeños archivos:

```
# echo domain magma.com.ni > /etc/resolv.conf
# echo nameserver 165.98.128.2 >> /etc/resolv.conf
```

crea (en caso que no existe) un archivo `/etc/resolv.conf` con el contenido:

```
domain magma.com.ni
nameserver 165.98.128.2
```

el uso de `>` y `>>` en la primera y segunda línea es a proposito. Lo primero sobreescribe el contenido actual del archivo con la salida del comando anterior y lo segundo – >>– anexa el texto ubicado después de `echo` al final del archivo.

`echo` no es un programa propio sino un comando internto del shell.

`cat - >` *archivo* Vacia *archivo*, en caso ya existe, si no, crea un archivo vacío con el nombre *archivo*. Después lee todas las teclas que se pulsan y los anexa uno por uno a este archivo. Al pulsar `[Ctrl]`+`[D]` termina la captura y se cierra el archivo. Normalmente se puede corregir errores dentro de una línea y mover el cursos con las teclas de cursor `[Izquierda]` y `[Derecha]`.

`cat` *archivo* copia el archivo nombrado en la línea de comando a la pantalla, y sirve de esta manera para visualizar su contenido. Si el archivo es más largo que la cantidad de líneas en la pantalla no se podría visualizarlo de manera cómoda. Para visualizar y navegar archivos grandes existen otros comandos avanzados, vea el capítulo E al respecto. Pero por velocidad y para archivos pequeños `cat` es justamente apropiado.

`cat` *cabeza cuerpo firma* `> carta.txt` con**cat**ena tres archivos, que – presuntamente – contienen un encabezado preparado, un texto y las cláusulas finales para concluir amablemente una carta, que es, en este ejemplo grabado en el archivo `carta.txt`. Este ejemplo muestra otro uso de `cat`, que permite ahorrar tiempo de tipeo. *cabeza* y *firma* pueden crearse una sola vez, y uno tiene que crear únicamente el contenido – *cuerpo* – para cada correo. Inclusive, puede usarse '`-`' en el lugar de *cuerpo* y crear una carta completa al instante, introduciendo el contenido manualmente, línea por línea, y terminado con `[Ctrl]`+`[D]`

### 4.5.5. grep

Si buscamos un archivo que contiene una cierta cadena de caracteres podemos usar este comando de la siguiente manera:

```
$ grep -r -l cadena *
```

La salida del comando es una lista de cero o mas líneas que contienen uno por uno los nombres de archivos en el directorio actual y sus subdirectorios (`-r` ...recursivo).

Otro uso de grep es, imprimir todas las líneas de un archivo, que concuerdan con una cierta cadena de caracteres.

```
$ grep cadena archivo
```

O por otro lado, imprimir todas las líneas que *no* contienen la cadena:

```
$ grep -v cadena archivo
```

un ejemplo práctivo para lo último es para leer archivos de configuración. En estos, muchas veces hay innumerables líneas de comentario, que comienzan con el caracter #.

```
$ grep -v '^#' archivo
```

lee el archivo con el nombre *archivo*, y lo repite en la pantalla, oprimiendo las líneas que comienzan con #. Las comillas sencillas previenen que el shell interprete por si mismo la parte "' *archivo*" como comentario y lo omite, y el circumflejo ^ simboliza el comienzo de la línea. Este símbolo previene de las expresiones regulares, que son explicadas prevemente en el capítulo G. La cadena de búsqueda no solamente puede ser una cadena fija de caracteres sino una expresión regular

### 4.5.6. sed

El nombre `sed` es una abreviación de '**s**tream **ed**itor'. En vez de editar interactivamente un archivo, sed lee línea por línea un archivo de la línea de comandos o desde el teclado, y aplica comandos de edición de texto, como reemplazar una cadena por otra, borrar o insertar cadenas de caracteres etc. Para ubicar en qué parte del texto aplicar los comandos, se usan expresiones regulares. El síntaxis común de una invocación de `sed` puede ser:

```
$ sed líneacomandos archivo
```

La expresión *linea* selecciona las líneas a las cuales se refieren los *comandos*, formas comunes para *línea son*:

*número* selecciona la línea que corresponde a *número* en el archivo

*primero*~*salto* selecciona la línea que corresponde al número *primero*, despues a *primero+salto*, después *primero+2×salto*, y succesivos.

*/regex/* selecciona todas las lineas que concuerdan con la expresión regular *regrex*.

*comienzo*,*final* selecciona el rango entre la primer línea seleccionada poar *comienzo* y la subsiguiente línea que concuerda con *final*.

Los comandos de edición consisten de una sola letra, seguida eventualmente por comentarios. Varios comandos pueden ser concatenados con '`;`' y grupos de comandos se encierran entre llaves: `{}`. Algunos de los comandos son:

`s/`*busca*`/`*reemplaza*`/`*opciones* la cadena de caracteres *busca* es sustituida por la cadena *reemplaza* una vez por cada línea. Si busca no se encuentra en la línea, no se efectua el reemplazo. Entre las opciones hay:

`g` buscar y reemplazar las cadenas en toda la línea.

`p` imprime la línea, siempre y cuando se haya efectuada al menos una sustitución

`p` imprime la línea actual – se usa comunmente con el comando `sed -n`

`d` borrar la línea actual.

`N` Concatena la siguiente línea con la actual. Eso no elimina el caractér final de línea entre la primera y la segunda línea.

`n` descarta la línea actual y lee la siguiente.

`sed -n` No imprime por omisión las líneas leídas en la entrada.

Es tiempo para algunos ejemplos. Si ha aplicado el ejemplo para eliminar las líneas de comentarios de un archivo del capítulo anterior, tal vez se hay dado cuenta, que hay archivos de configuración que contienen también muchas líneas vacías. Estos se pueden eliminar juntos con los de comentario con:

```
$ grep -v '^$' archivo | grep -v '^#'
```

La expresión regular `^$` es: comienzo de línea seguido por final de línea – una línea vacía. Con la concatenación de dos expresiones regulares por el operador lógico 'OR' – | – la barra vertical podemos combinar la selección de líneas a suprimir, pero grep no hará lo correcto (intentalo). `sed` nos ayuda:

```
$ sed '/^#\|^$/d' archivo
```

En palabras profanas: lee el archivo *archivo* línea por línea, y copialos a la salida – pero: si encuentras una línea que comienza con # o que está vacía, borrala.

Otro ejemplo, para solo imprimir cada segunda línea de un archivo:

```
$ sed -n 2~2p archivo
```

Concatena cada segunda línea con la primera, separada por espacio-guión-espacio:

```
$ sed -n '1~2{N;s/\n/ - /p;}'
```

Imprimir un archivo a partir de una línea que contiene "estimados señores":

```
$ sed 1,/estimados señores/d archivo
```

Imprimir un archivo a partir e inclusive esta misma línea:

```
$ sed -n '/estimados señores/,$p' archivo
```

Es sencillo modificar archivos existentes:

```
$ sed s/Ana/Eva/g carta_ana.txt > carta_eva.txt
```

sustituye en el archivo `carta_ana.txt` todas las ocurrencias de "Ana" por "Eva", y escribe el resultado en el archivo (nuevo) `carta_eva.txt`. Notemos, que no se puede sustituir un archivo en si mismo, sino tiene que escribirse en un nuevo archivo, y eventualmente renombrarse posteriormente.

Una aplicación más amplia de este último ejemplo es escribir un archivo plantilla, y crear de este automáticamente configuraciones individualizados. Para el siguiente ejemplo regresamos a la administración de sistemas. Supongamos que tenemos que crear archivos de configuración individuales para dominios Web virtuales. Elaboramos un archivo plantilla, `virtualdomain`:

```
<VirtualHost NRO_IP>
  ServerAdmin webmaster@DOMINIO
  DocumentRoot /var/www/DOMINIO
  ServerName www.DOMINIO
</VirtualHost>
```

Cada dominio virtual requiere que especifiquemos el número IP y el nombre del dominio, un ejemplo: 165.98.128.4 y linuxbook.org.ni.

```
# sed s/NRO_IP/165.98.128.4/\;s/DOMINIO/linuxbook.org.ni/ virtualdomain > /etc/apache/linuxbo
```

Crea el archivo de configuración `/etc/apache/linuxbook.org.ni` con el siguiente contenido:

```
<VirtualHost 165.98.128.4>
  ServerAdmin webmaster@linuxbook.org.ni
  DocumentRoot /var/www/linuxbook.org.ni
  ServerName www.linuxbook.org.ni
</VirtualHost>
```

### 4.5.7. Editores de texto

No pretendemos dar un tutorial para cada uno de los editores de texto en esta sección. Hay fichas de referencia para algunos de ellos en el anexo. Primero explicamos en breve su función y después les damos a conocer los editores más comunes en Linux.

Un editor de texto se distingue esencialmente de un procesador de palabras. Mientras el segundo comúnmente utiliza un formato de archivo especializado y se orienta en la presentación del texto, los editores de texto, o editores ASCII, son orientados a la creación de archivos de "texto plano". Un archivo de texto plano es una concatenación de líneas, compuestas por caractéres, normalmente en código ASCII, separados por la secuencia

*final de línea* – EOL, end of line – en Unix [Ctrl]+[J] - el símbolo para avance de línea – LF, linefeed – correspondiente al número hexadecimal 0x0a. Para introducir esta secuencia se utiliza la tecla [Enter]. Cabe mencionar, que en los sistemas operativos de la empresa Microsoft (Dos, Windows) se utiliza una secuencia [Ctrl]+[M] [Ctrl]+[J], para indicar EOL, y en los sistemas de Apple se utiliza [Ctrl]+[M]. [Ctrl]+[M] es el símbolo para el retorno al comienzo de la línea – CR carriage return – correspondiente al número hexadecimal 0x0d. Hay programas para convertir todo un archivo de texto entre uno y otro formato de final de línea (p.ej `recode`) y algunos de los editores reconocen el formato de fín de línea y se adaptan a el, o lo convierten según necesario. Un archivo de texto plano por lo general se puede copiar directamente a la impresora, la cual imprime caracter por caracter y – si es configurada correctamente – realiza correctamente los saltos de línea. No se necesita una conversión del archivo al formato de impresión.

Los archivos de texto plano tienen una importancia grande en la programación y para los archivos de configuración: los programas compilados y scripts por lo general son escritos en forma de archivos de texto plano, igual como los archivos en los cuales se definen los parametros y configuración del sistema operativo y de las aplicaciones que se ejecutan en ello.

En la edición de archivos a menudo se encuentran dos dificultades para principiantes: primero, el reconocer de que una o varias "líneas vacías" no son nada mas que una repetición de la secuencia final de línea, que es insertada como un simple caracter y que también es borrada como un simple caractér, de hecho, es (en Unix y Macintosh) una secuencia de simples caractéres. La segunda dificultad es acerca del cursor, o punto de inserción y la eliminación de un caracter del archivo. Al visualizar el texto, muchas veces se mira un caracter resaltado, o una linea vertical entre dos caracteres. Este es el punto de inserción: el siguiente caracter que se pulsa en el teclado se *anexa después* del caracter que está a la izquierda del punto de inserción. En los terminales de texto, donde el punto de inserción es marcado con un rectángulo iluminado en el caracter "actual", el punto de inserción es a la *izquierda* de este, entre el caractér marcado y el caracter próxima a la izquierda. Hay dos teclas para borrar un caractér: [Del] elimina el caractér a la *derecha* del punto de inserción, y [BkSp] o [←] en la parte superior derecha del teclado elimina el caractér a la izquierda del punto de inserción.

El editor tradicional de los unices es:

`vi` - sinónimo para *visual editor*, es basado en `ex`, un editor simple que manipula un archivo línea por línea. `vi` manipula el punto de inserción y el texto con comandos que consisten en un solo caracter: [x] para borrar un caracter, [j] para mover el punto de inserción a la izquierda, [/] para búsqueda, [s] para búsqueda y reemplazo etc. Estos comandos se encuentran en muchos otros programas de unix. Una carta de referencia se encuentra en el anexo C.

El editor oficial del proyecto GNU es

**Emacs.** Emacs no solo es un editor sino un ambiente de trabajo completo con capacidades y extensiones inmensos. Igual como vi utiliza teclas para su manejo, pero utiliza

como prefijo las teclas `[Alt]` y `[Ctrl]`, ambos editores pueden ser usados hasta con los teclados más sencillos, sin teclas de cursos y con terminales (monitores) con mínimas capacidades de manipulación de la pantalla. Por eso son muy práctico en la edición de textos en computadoras remota, donde a veces el método de conexión impide el uso de caracteres especiales o efectos avanzados en la pantalla. Las versiones Emacs para el ambiente gráfico también son manejados por menús, el ratón, y por lo general todas las teclas de movimiento de cursor y el borrado de caracteres son habilitados. En concepto de comparación con `vi` he aqui los comandos respectivos: `[Ctrl]+[d]` (delete) borra un caracter, `[Ctrl]+[b]` (back) mueve el punto de inserción a la izquierda, `[Ctrl]+[s]` (search) inicia la búsqueda, `[Esc] [%]` inicia búsqueda y reemplazo. La carta de referencia para emacs está en B.

Otros editores son:

**jove** mini editor con semejancias a emacs

**jed** otro editor similar a emacs de escala media y muy flexible

**nano** una reimplementación de licencia libre del editor pico, que es muy conocido por usuarios del programa famoso de correo electrónico pine.

**mcedit** el editor integrado en Midnight Commander, clonado del Norton Commander para MS-Dos.

Todos estos editores trabajan en ambiente de texto, algunos también en ambiente gráfico y con soporte para el ratón.

### 4.5.8. stdin/stdout/stderr

En 4.5.4 ya nos hemos encontrado con los operadores `>` y `>>`, en 4.5.6 con `|`. En esta sección vamos a profundizar un poco sobre los conceptos detrás de ellos.

Los comandos usados en los shell de unix pueden considerarse en la mayoría de los casos como *filtros*, parecidos a filtros o procesadores de agua. Ellos reciben una serie de caracteres – stream = corriente, en su entrada, los manipulan y remiten el resultado a la salida. Si surgen iregularidades, por ejemplo no hay suficiente espacio en el disco duro para el procesamiento, se suele dar al operador un mensaje respectivo. Estos mensajes se transmiten por una salida aparte, llamada la salida estándar de error – stderr. La entrada y salida estándar se llaman stdin y stdout respectivamente. En el shell interactivo stdout y stderr por defecto son conectados a un canal de entrada/salida que transmite los caracteres producidos a la pantalla y el stdin esta conectado a un canal que recibe los caracteres que pulsamos en el teclado. Notese, que de esta manera los mensajes de errores y avisos sobre el procesamiento efectuado por un programa se intermezclan con la salida del programa.

Los operadores arriba introducidos sirven para reconectar las intradas y salidas de los programas a canales de entrada/salida diferentes, antes de ejecutarlo.

*canal* un canal de entrada salida puede ser (entre otros) una archivo común, una dispositivo de entrada salida, como la pantalla o una impresora, o inclusive un dispositivo de almacenamiento como un disco duro, o una partición específica de el.

`>` *canal* esta secuencia indica, que la salida estándard del programa, indicado por el ángulo abierto, se redirige al *canal* indicado.

`<` *canal* indica, que la entrada del programa es reconectado de su fuente original (el teclado...) al *canal* indicado

`2>` *canal* indica redirección de la salida estándar de error al *canal* indicado.

Pueden redirigirse las entradas y salidas en cualqier combinación. Algunos ejemplos:

```
$ grep -r hda /etc/* 2> /dev/null
```

Este comando resultaría en muchos mensajes de error cuando intenta recursivamente entrar en subdirectorios de /etc/ que son protegidos contra lectura de usuarios comunes. La redirección envía todos los mensajes a `/dev/null` que es un canal de entrada salida que descarta cualquier caracter que es enviado hacia el. Muchas veces se refiere a `/dev/null` como <u>bitbucket</u>, o sea, el basurero de bits.

```
# X -probeonly 2> test.log
```

Ejecuta el gestor gráfico `X` del ambiente gráfico X Window System en modo de prueba, y acumula los resultados, que son escritos en la salida estándar de error en el archivo `test.log`. Eso es útil, ya que este programa emite una grán cantidad de diagnósticos sobre memoria, tarjeta gráfica detectada y disposición del monitor, que no calzan en una sola pantalla. Posteriormente el archivo test.log puede ser visualizado con un editor o un visualizador (vea E).

```
# tai64nlocal < /var/log/qmail-send/current > qmail.log
```

Lee el primer archivo, lo filtra a través del programa `tai64nlocal` que convierte sellos de tiempo en formato 'temps atomiqué internacional' al formato del horario local, y escribe el resultado al archivo `qmail.log` en el directorio actual.

```
$ </etc/apache/httpd.conf   sed '/^#/d'   >apache.config
```

Imprime en la pantalla las líneas del archivo `/etc/apache/httpd.conf` que no comienzan con #, el caracter que introduce un comentario. Este ejemplo ilustra, que no es preciso, que la redirección se indique después del comando, también puede ubicarse en frente, y que no se require introducir un espacio en blanco entre el operador de redireccionamiento y el canal.

*comando*1 `|` *comando2* el símbolo `|` es diferente al redireccionamiento común en el sentido que crea en el acto un dispositivo llamado pipe – tubo. La salida estándar de *comando1* es redireccionado a la entrada estándar de *comando2*. De esta manera se concatena el procesamiento del primer filtro con el segundo. Es característico, que al menos durante el tiempo que *comando1* produce caracteres en su salida estándar los dos comandos se ejecutan en parallelo. Pueden concatenarse una cantidad arbitraria de comandos mediante pipes, obteniendo resultados muy sofisticados con los comandos sencillos de manipulación de archivos que trae la "caja de herramientas" de unix.

```
$ grep -v  '<Directory' /etc/apache/httpd.conf | grep -v '</Directory' | grep -v '^#' |less
```

lee el archivo de configuración del servidor web apache, y elimina con el primer `grep` todas las líneas que contienen *<Directory*, con el segundo `grep` todas las líneas que contienen su contraparte *</Directory*, y con el último `grep` todas las líneas de solo comentario. El resultado final de este filtraje es visualizado con el paginador `less`. Tal vez esto no parece un ejemplo muy convincente, pero uno/a puede verse en una situación que quiere filtrar información de un archivo, y concatena de esta manera en un proceso de intento y error los filtros, hasta obtener un resultado conveniente. También mostramos en este ejemplo, que igual como con `>`, `<` y `>>` no es necesario interponer un espacio en blanco entre el operador pipe y sus operandes.

# 5. Manejo de Linux en ambiente gráfico




Los sistemas operativos Unix han tenido la fama de forzar el/la usuairi@ a usar comandos crípticos desde la línea de comando. Mientras siempre queda abierta esta opción, hoy en día no hay necesidad para un@ usuari@ común de temer el uso de una computadora con estos sistemas. El escritorio virtual se ha hecho un estándar en las computadoras Unix, y la mayoría de las aplicaciones profesionales y/más difundidos existen en versiones para diferentes variantes de Unix, permitiendo al/la usuari@ cambiar de sistema operativo e inclusive de computadora con la posibilidad de utilizar los mismos datos y la misma forma de trabajar en cualquiera de ellos[1].

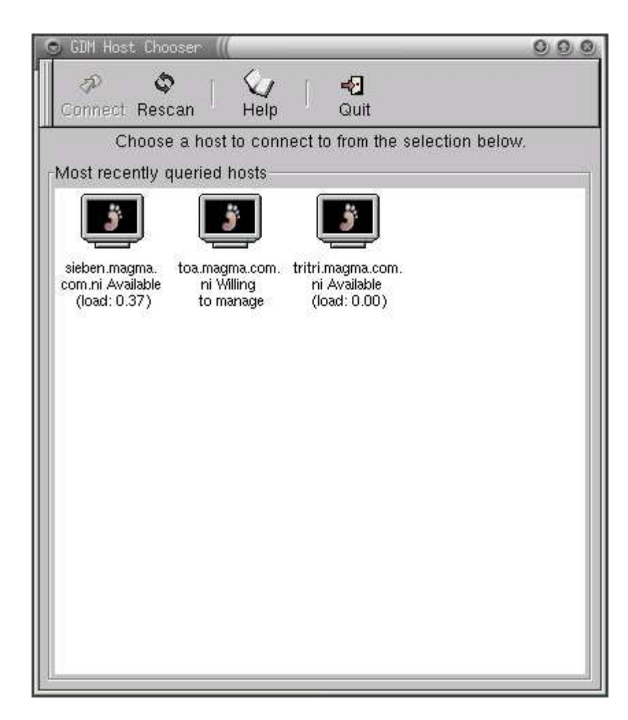
Esto fue posible en gran parte por una triada: la cuasi estandardización de Unix, la disponibilidad del servidor gráfico X-Windows, y el desarrollo de los sistemas de Escritorios virtuales - Desktop.

Los Escritorios virtuales han vivido una proceso de maduración desde su percepción, en donde el manejo y la forma de presentación se ha perfilado y su uso ha alcanzado una gran aceptación y asimilación. Con la creación de programas Desktop que se adhieren a este perfíl se permite el manejo fácil de cualquier persona familiarizada con este estilo de trabajo, sin la necesidad de una capacitación específica a cerca del uso de la computadora. En este capítulo se presentará algunos elementos muy básicos del uso del escritorio Gnome Desktop, pero también ya se presenta el contenido en una estructuración que refleja la

[1] Programas y sistemas elaborados de esta forma se llaman *interoperables.*

diferenciación de funciones de Interfáz, que hace los escritorios virtuales de Unix tan flexibles (y variados).

## 5.1. Chooser y Login Widget




En primer lugar hay que notar, que el ambiente gráfico en los sistemas Unix preservan las caracteristicas de seguridad que tiene el ambiente textual. Para acceder a una "sesión" gráfica hay que pasar por un proceso de autentificación al sistema. Esto se realiza a través de un Login Widget[2], que presenta un campo de entrada textual para el nombre de usuario, y otro para la clave de autentificación. Si el/la usuari@ se autentifica correctamente, el sistema permite la entrada a una sessión en la computadora.

X Windows, sin embargo es altamente integrado en el ambiente de red, de tal manera, que una sesión de usuario no necesariamente tiene que ejecutarse en la computadora local (donde está sentado el/la usuari@) sino puede efectuarse también en una computadora remota. Esto es parecido a una sesión telnet en el ambiente textual. Sin embargo necesitamos un mecanismo de conexión a esta computadora remota. En computadoras donde está habilitado el acceso a máquinas remotas aparece primero un selector de computadoras, llamado "choosee", una ventana que presenta una lista de máquinas disponibles. El/la usuari@ selecciona con el ratón una de las computadoras (con doble-click), la cual a continuación presenta el Login Widget ofreciendo una sessión (remota) de la manera arriba descrita.

[2]Widget es "cosa" en inglés, en los ambientes gráficos de computación significa cualquier elemento gráfico de interacción con el usuario: ventanas, botones, etc.




Nota: Las dos gráficas presentadas corresponden a los Widget Chooser y Login del Gnome Desktop Versión Ximian. Hay numerosas diferentes representaciones de Chooser y Login y las versiones básicas proveidos con X-Windows, aunque son menos llamativos todavía tienen la mayor rapidéz y estabilidad.

## 5.2. Gnome Desktop

El "Desktop" o "Escritorio" es el conjunto de elementos de control e interacción y sus funciones que permiten a un usuario organizar el trabajo con programas y aplicaciones en un ambiente gráfico. Esto consiste en elementos de organisación de elementos visuales (Window Manager), manejo de machotes de organización (Session Manager), Elementos gráficos de control del sistema (Lanzadores, acceso a dispositivos, indicadores, visualizadores, etc.).

El proyecto Gnome implementa un Desktop completo, altamente flexible y configurable.

Existen otros Desktops, como p.e. el KDE-Desktop.

No es necesario trabajar con un Desktop "completo" para poder utilizar programas gráficos en X-Windows.

Los Elementos más importantes en el Gnome Desktop son:

**Panel** es una barra, ubicada generalmente en la parte inferior de la pantalla, donde se ubican iconos, llamados appletts, que activan funciones con "un solo click", o visualizan información del sistema o del estado del Desktop o de una aplicación. El elemento más importante es el "Menu Principal" o "Main Menu", una huella de pie en forma de 'G' frente a un trasfondo gris. Apretando este icono se desprende el menu que da acceso a casi toda la funcionalidad.

Otros iconos, llamados "lanzadores" o "launcher" pueden crearse libremente. Si se hace Click en ellos, lanzan (arrancan) el programa asociado con ellos.



**Gmc** Es un manejador de archivos, con el cual se pueden manipular dispositivos de almacenaje como discos duros, floppys, cdrom (montar, desmontar, etc.); directorios y archivos (borrar, mover, copiar, renombrar, visualizar). Gmc trabaja como browser (navegador) de archivos en una o mas ventanas propias. Puede manipular enlaces en forma de "URL", que le permite lanzar aplicaciones en dependencia del tipo de archivo, segun la asociación del tipo "MIME" del archivo. Enlaces en el Escritorio mismo se visualizan en forma de iconos en la pantalla para su inmediata manipulación con el ratón.

**Control ▷ SectionX-Windows**

X es una aplicación que nos permite utilizar programas con interfaz gráfico, quiere decir, que los comandos para operar los programas se pueden dar via movimientos del ratón, apretando las teclas del ratón, sin embargo también con el teclado. Los resultados obtenido de los programas se visualizan en fórma gráfica, p.e. con cambios de colores o movimientos de objetos gráficos en la pantalla.

X-Windows no interfiere en el proceso de interacción entre el usuario y los programas de aplicación, sino más bien provee un ambiente estandardizado para los programadores. X trabaja en forma Servidor - Cliente. En la computadora donde "estamos sentados" se ejecuta el "Servidor X", que recibe nuestros señales y los transmite a los programas de aplicación - los clientes. También recibe mensajes desde las aplicaciónes, las cuales convierte en las figuras gráficas correspondientes. De esta forma libera a los programas de aplicación de tener conocimiento sobre la forma de programar el adaptador gráfico de la computadora donde son utilizados y un programa escrito para X Windows puede funcionar en cualquier computadora o sistema operativo, siempre y cuando tiene acceso a un Servidor X. Esta forma de funcionamiento tiene otra ventaja, ya que no tienen que residir el programa de aplicación en la misma computadora como el Servidor X. Eso nos permite acceder a nuestro escritorio virtual remotamente, via una red local e inclusive desde el Internet.

Para el usuario la validéz de un sistema consiste en gran parte en la disponibildad de programas de aplicación. X Windows permite la ejecución de cualquier programa Unix textual a través de emuladores de terminales textuales. Estos son parecidos a la ventana DOS en MS-Windows, pero con la diferencia que este provee una simulación del ambiente de *ejecución* (no todos los programas funcionan), mientras un terminal texto en X-Windows provee al programa Unix una simulación del ambiente de *entrada/salida*, y la ejecución queda inerta.

Sin embargo, el uso de programas textuales se está reduciendo con la disponibilidad de una vasta gama de aplicaciones gráficas, especialmente con la venida de los Desktop KDE y Gnome, que se basan en poderosas y modernas herramienta de programación y permiten la inclusión del código original de programas textuales en ambientes gráficos.

## 5.3. Generalidades de GUI's

### 5.3.1. Ratón

Generalmente se observa en la pantalla un objeto pequeño, llamado cursor o puntero, que cambia su posición con el movimiento del ratón de la computadora. La forma y o el color del cursor pueden cambiár para dar retroalimentación al usuario. p.e. si es una flecha o una cruz se supone que el usuario está posicionando el cursor, dentro de áreas destinado para textos es una línea vertical en forma de 'I', si es una mano indica que se puede hacer una operación, muchas veces de cambio de contexto.

Apretando las teclas del ratón solo por un instante se llama "hacer Click", por el sonido que emite la tecla. Haciendo Click dos veces en rápida successión se llama "Doble Click"; tres veces: "Triple Click". Esto es opuesto a apretar el botón y no soltarlo y es interpretado diferentemente. Mantener apretado el botón mientras se mueve el cursor se llama "jalar", o "Drag". ¡Ojo! Generalmente se aprieta – mueve – suelta. Se dice "soltar" o "Drop" a la acción de finalmente soltar el botón después del movimiento.

Existen ratónes con solo un botón (MacIntosh), en PC's compatibles se utilizan generalmente ratones con dos botones, en X-windows es muy común el uso de ratones de tres botones. Existen ratones con una rueda de "navegación" y ratones con cuatro o más botones.

La cantidad de posibles combinaciones de Click, apretones y movimientos aumenta con la cantidad de botones, lo que permite usar el ratón como dispositivo de comando mas versátil, pero a la vez se necesita más aprendizaje y práctica para manejar un programa que utiliza combinaciones complejas de comandos de ratón.

En MS-windows el Click izquierda (botón izquierda) activa el elemento que se encuentra actualmente bajo el cursor - se dice que este elemento gana el foco "gain the focus", o: a partir de este momento los demas comandos, especialmente del teclado se dirigen hacia el elemento.

El doble Click izquierda "abre" el elemento, o sea activa el comando más frecuentemente asociado con el elemento.

El Click derecha abre un menu colgante del cursor, llamado "menu contextual", que prove una lista de los comandos que pueden operar en este.

En X-Windows estas reglas no son estandardizados, por lo que hay mucha variedad. Por lo general el Click y Doble Click izquierda tienen la misma función de "activar" el elemento, el Click o apretar derecha abre un menu contextual, y el apretar el botón mediano muchas veces prove funciones de movimiento del elemento selecionado.

### 5.3.2. Ventanas

Los elementos mostrados en la pantalla son figuras geométricas, generalmente rectángulos que representan "areas de control" de diferentes programas y sirven para la interactuación con el usuario de estos mismos.

En MS-Windows se llaman Ventanas, en X-Windows muchas veces se llaman "widgets", que tiene significado de: útil, dispositivo de entrada.

X-Windows prove a los programas áreas rectangulares que pueden usarse para información gráfica y/o textual. La ubicación y el tamaño de este área es determinado por el "administrador de ventanas", un programa que dibuja un "marco" alrededor del área, y permite al usuario mover, y redimensionar, esta ventana a través del mouse. También se puede "minimizar" o "iconificar" la ventana, reduciendola a un símbolo e inhibiendo interacción, hasta que se vuelva a "reestablecer" a su tamaño anterior, o se "maximisa", lo que es darle a la ventana el tamaño máximo posible en la pantalla, tapando todos los demás objetos.

"Enrollar", "shade" una ventana, es mantener su barra superior en lugar y tamaño, mientras se elimina graficamente el resto de su contenido. La operación contraria es "unshade".

Mientras la maximización en MS-Windows hace actuar la ventana de una manera especial, p.e. no se puede cambiar su tamaño, en X-Windows muchas veces solamente es una operación que redimensiona el tamaño al máximo. Puede haber máximización horizontal (ancho), vertical (alto) y ambos.

Otra forma de modificación automática de tamaño es "fill" o llenado, donde se respetan ciertos otros elementos para no taparlos.

Para modificar el tamaño de una ventana interactivamente se "jala" de una de las esquinas, o de un borde de la ventana hacia la dimensión deseada.

También a través del marco y con comandos del ratón se puede terminar normalmente o esforzadamente el programa que "posee" la ventana, estos comandos a veces se llaman "destruir", y "anihilar", o "terminate" y "kill" respectivamente. En vez de destruir tambien se lee "delete".

Pueden haber varias ventanas e inevitablemente uno se posiciona "encima" de otro cuando el espacio ya no alcanza para mostrarlas todas a la vez. Se establece un orden de "bottom" que es lo "mas a fondo" a "top" que es la ventana en primer plano que es completamente visible. "Raise" una ventana es subirla un "nivel", "Lower" es bajarla. "Raise to top" es subirla a primer plano, mientras "send to bottom" es ponerla "abajo" en la "pila".

### 5.3.3. Gestores de Ventanas

Para el manejo de las ventanas existe una gran variedad de formas y convenciones según el sistema operativo: McIntosh, NeXt-Step, MS-Windows, X-Windows "tradicional".

Las últimas tendencias en la programación en X-windows proveen administradores (gestores) de ventana cada vez más sofisticados, y configurables.

Mientras hace poco la noción ha sido la de modificar un gestor de ventana existente de tal manera que provea un "look&feel" - "mirar y sentir", parecido a un determinado sistema, actualmente se diseñan nuevos gestores, que son capazes de emular diferentes sistemas desde un inicio.

En efecto disponemos de una gran variedad de gestores de ventanas, desde los tradicionales hasta los más sofisticados. A continuación una lista de gestores en aproximadamente este orden, que a la vez representa "velocidad de operación" decresciente:

**twm** Tab Window Manager. Es importante, porque "siempre" funciona y se usa como último recurso si otros gestores fallan o no son presentes. Pequeño y rápido. El botón izquierda en el desktrop abre un menu de programas, el botón deracha operaciónes del gestor y la salida de X, y el botón en el centro operaciones en ventanas (kill, move, etc.) – (¡Verificar esta información!).

**mwm** Motiv Window Manager. Motif es un ambiente de programación comercial muy popular. Una versión libre de Motif es LessTiff.

**fvwm** Free Virtual Window Manager. Es un succesor de twm. Utiliza menos memoria durante su corrida e introduce el concepto de escritorios virtuales que son conjuntos de ventanas. Cambiar a otro desktop significa visualizar en el monitor otro conjunto. Se pueden configurar muchos diferentes conjunto y sin salir de X trabajar así en varios diferentes "ambientes".

**fvwm2/95** Modificaciones de fvwm para aparentarse a MS-Windows-95.

**AfterStep** Simula el Desktop NeXT-Step, extremadamente elegante. NeXT fue una generación de computadoras diseñados por una empresa derivada de MacIntosh.

**ICEwm** Gestor bonito y flexible con barra de tareas/sistemas.

**Enlightenment** Gestor principal del Gnome Desktop. Configurable con Scheme (Lisp).

**WindowMaker** Gestor preconfigurado de Debian Linux. Tiene una presentación y un manejo alternativo a los gestores parecidos con MS-Windows.

**Sawmill** También configurable con Scheme. Mi preferencia porque es más rápido que Enligthenment y muy versátil.

**qvwm** Gestor que simula Windows 95 y ocupa muy pocos recursos.

**wm2** Un gestor definitivamente mínimo.

### 5.3.4. Interacción con el usuario

**Follow\-toFocus** hay que hacer click, o inclusive doble click para una ventana reciba el focus.

**RaiseDescriptionManualDescriptionStackDescriptionrandom** se escoje un lugar arbitrario para la nueva ventana.

**firstDescriptioninteractive** lo mismo como Manual Placement

**Animate** se agrega efectos gráficos, a los transformaciones de ventanas. p.e. al crear una ventana se deja aparecer de alguna orilla de la pantalla de donde se desplaza visiblemente hasta llegar a su destino final. O al minimizar y restablecer se simbolisa que la ventana se encoge y estira respectivamente.

# 6. Scripts

Con script (escrito) se denomina programas que se escriben en texto claro y que son interpretados directamente, o sea, que no tienen que pasar por un proceso de traducción. En otros ambientes se llaman también archivos batch. Los lenguajes script se distinguen en el sentido, que ellos mismos solo tienen la funcionalidad de aglutinar comandos "externos", los cuales pueden ser entonces de una complejidad arbitraria, sin que el lenguaje script mismo tenga muchos comandos. Existe un sinnúmero de lenguajes script, algunos ejemplos son Tcl, perl y la gran variedad de shell's (interpretadores de comandos) como son zsh, csh, ksh, sh. El shell estándard del proyecto GNU y de Linux es bash e incorpora un lenguaje script muy poderoso.

Podemos ver un script como una listas de comandos que el sistema linux permite ejecutar automáticamente, en vez de tener que introducirlos manualmente. Hay tres conceptos que se asocian con los lenguajes script, o la programación shell: la creación de "macros", o sea subsumir bajo un solo nuevo nombre de comando una seríe de commandos individuales, la realización de tareas complejas a través de la combinación de varias tareas sencillas - divide and conquer, y la realización de tareas diferentes pero parecidas con un solo comando - parámetros.

Para crear un script se anotan los comandos individuales en secuencia en un archivo ascii, el cual se puede interpretar a continuación, utilizando el nombre del archivo como comando shell. Para esto se le necesita dar permiso de ejecución al archivo, ej. ("macro"):

Escriba en un archivo de nombre 'archivos' la línea: "ls", ejecute el commando: "chmod +x archivos", y a continuación "./archivos"

El comando 'ls' se ejecutara.

Vamos a ver este proceso más detallado:

1. Bash reconoce, que "./archivos" no es un comando interno.
2. Normalmente busca en la via de acceso (Variable PATH) si encuentra el archivo. En este caso, el nombre del archivo está especificado absolutamente ('./' es el directorio actual, y 'archivos' existe) entonces lo trata de ejecutar.
3. El atributo de ejecución está activado entonces se determina si el archivo es en uno de los formatos binarios que se pueden ejecutar directamente. En este caso es un archivo de texto, no 'binario'
4. Si la primera línea tiene la secuencia "#!" seguido por el nombre de un comando (filtro), este comando se ejecuta, usando el archivo script como entrada en stdin.
5. En nuestro caso no existe esta línea, por lo que se ejecuta el shell por omisión (default shell) como filtro.

6. Este nuevo bash (sub-shell) lee la línea 'ls', y lo interpreta como si hubieramos typeado el comando a mano: Se ejecuta 'ls' y los archivos del directorio actual se alistan en la pantalla.

7. El nuevo bash termina y regresamos al prompt del bash originador.

## 6.1. Gramatica para bash

**#** inicia un comentario, el resto de la línea se ignora

**$** inicia una sustitución de variable
seguido por un número 0 - 9 reproduce en su lugar el texto del 'token' 0 - 9 respectivamente en la línea de commando
seguido por un texto reproduce el contenido de una variable shell
seguido por una expresión reproduce el texto del stdout de la expresión, ejecutandola en un sub-shell.

**** preserva el siguiente carácter si este es especial, o introduce caracteres especiales, \\\\ se convierte en \\, se llama "escape caracter".

**\n** nueva línea (newline) ; '\\n' se convierte en '\n'

**\r** retorno al inicio de la línea

**\t** tabulador

**\g** suena el timbre del terminal

**'** (quote, single quote), protege una secuencia de caracteres y palabras del procesamiento de línea de bash; 'au\nque no' se mantiene intacto, inclusive '\n'

**"** (double-quote) igual como el single quote, pero "\" y "$" y ' preservan su función, "\" solamente si es seguido por $, ', ", \, or <newline>.

**.StandardVariables** bash, son nombres que pueden tener asignados cadenas de caracteres arbitrarios. Comunmente se utilizan letras mayusculas, para mas fácil diferenciarlos de los comandos.

Variables pueden ser leídos y escritos en cualquier momento. La preconfiguración de la computadora asigna un juego completo a diferentes variables tradicionalmente usados en unix. El juego de este variables permite detectar desde los programas y comandos en que estado y con que usuario etc. se encuentra el sistema en cualquier instanto. Porque definen el ambiente de ejecución las variables también se llaman "variables de ambiente" o "ambiente" (Environment variables, Environment).

El comando interno de bash 'set' escribe en stdout todas las variables de ambiente y su valor. Para definir una variable simplemente se le asigna un valor:

```
VARIABLE=''Vamos a la playa''
```

Las variables solamente son válidos para su ambiente. Si se ejecuta un sub-shell ya no son "visibles". Ej: "echo $VARIABLE" no imprime nada en stdout. Para hacer válida una variable se tiene que "exportar"; después de definir la variable se da el comando: "export nombre" con el nombre de la variable. De esta manera se pueden crear jerarquías de validéz de variables (scope). Las variables que solo los usamos temporalmente son "protegidos" de modificaciones de sub-shells.

Bash ofrece una sintaxis simplificada: "export CAJA=523" es lo mismo como: "CAJA=523; export CAJA", sin embargo se prefiere utilizar la versión original en archivos script, para no crear incompatibilidad con 'sh'.

## 6.2. Decisiones

La vida real nos exige tomarlas. Bash incluye comandos, que nos permiten

1. Bifurcar
2. Iterar

Estos dos elementos se conocen como control de secuencias. La bifurcación se escribe de la siguiente manera:

```
if TEST-COMMANDS; then
    CONSEQUENT-COMMANDS;
[elif MORE-TEST-COMMANDS; then
   MORE-CONSEQUENTS;]
[else
    ALTERNATE-CONSEQUENTS;]
fi
```

Más especial:

```
if [ algunos comandos ]; then
   lista de acciones
else
   otra lista
fi
```

El código de salida (exit status) del último comando de 'algunos comandos' significa 'no válido' si es 0, en todos otros casos significa 'válido'. Cada comando (inclusive los script) returna un valor numérico de salida al shell que ejecuta. En el caso más simple el shell retorna 0 si el comando se ha encontrado, otro valor en caso que no. Todos los comandos y las funciones shell describen minuciosamente como se genera su exit status, de donde se puede deducir como utilizarlos para la bifurcación. El exit status del últio comando se tiene disponible en texto mediante la variable especial $?.

La iteración se escribe de la siguiente manera:

```
for VAR [in WORDS ...];
   do COMMANDS;
done
```

donde VAR es una variable shell creada en el acto, que se asigna en torno a cada cadena de caracteres presente en WORDS. si se omita se sustituye por $@, los parametros posicionales.

## 6.3. Programas shell resumidos

Veamos algunos ejemplos reales[1], que nos pueden servir como machotes en caso de tener que resolver una tarea parecida:

Para leer archivos "HOWTO" comprimidos:

```
#!/bin/sh
if [ "$1" = "" ]; then
   ls /usr/doc/faq/howto | less
else
   gunzip -c /usr/doc/faq/howto/$1-HOWTO.gz | less
fi
```

Remover recursivamente archivos temporales y de respaldo de diferentes aplicaciones conocidas y comprimir ciertos tipos de archivos:

```
#!/bin/sh
#SQUEEZE removes unnecessary files and compresses .tex and README files
#By Barry tolnas, tolnas@sun1.engr.utk.edu
# echo squeezing $PWD
find $PWD \( -name \*~ -or -name \*.o -or -name \*.log -or -name \*\#\) -exec rm -f
find $PWD \( -name \*.tex -or -name \*README\* -or -name \*readme\* \) -exec gzip -
```

Convertir nombres de archivos a menusculas:

```
for i in * ; do [ -f $i ] && mv -i $i `echo $i | tr '[A-Z]' '[a-z]'`; done;
```

Un script que hace lo mismo:

```
#!/bin/sh
# lowerit
# convert all file names in the current directory to lower case
# only operates on plain files--does not change the name of directories
```

[1] Los ejemplos en parte no funcionan porque dependen de una versión Linux específica. El adaptarlo a su sistema es ejercicio para el o la lector(a).

```
# will ask for verification before overwriting an existing file
for x in `ls`
   do
   if [ ! -f $x ]; then
      continue
    fi
   lc=`echo $x | tr '[A-Z]' '[a-z]'`
   if [ $lc != $x ]; then
       mv -i $x $lc
   fi
   done
```

Remover archivos core:

```
#!/bin/sh
 USAGE="$0 <directory> <message-file>"
 if [ $# != 2 ] ; then
    echo $USAGE exit
fi

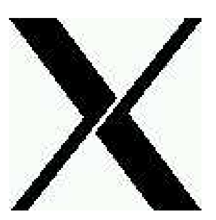
echo Deleting...
find $1 -name core -atime 7 -print -exec rm {} \;

echo e-mailing
for name in `find $1 -name core -exec ls -l {} \; | cut -c16-24`
do
   echo $name cat $2 | mail $name
done
```

# Parte II.

# Configuración

# 7. X-Windows



En este capítulo se discutan conceptos detraś del sistema X-Windows, que lo hacen ser muy distinto de otros GUI (Graphical User Interface - Interfáz Gráfico de Usuario). Cabe mencionar aquí otros sistemas de interfáz de usuario gráfico en el ámbito de las computadoras personales: MS-Windows con todas sus variantes y derivaciones, MacOS de Apple Computers, Plan9, NeXT-Step (ya no se produce), y finalmente X-Windows, que por su larga historia y su versatilidad se encuentra en uso desde computadoras personales, hasta en Workstations y computadoras grandes, en su mayoría con sistemas operativos de la familia Unix.

X-Windows no es parte del sistema operativo, sino una aplicación, que establece una "norma" (API) para el acceso a recursos hardware gráficos - la tarjeta de video, y de entrada - teclado, ratón, joystick, grafic-pads. Consiste de esta manera de un "servidor" gráfico, al cual pueden acceder programas "clientes" para interactuar con las usuarias. X-Windows tiene una alta integración en el ambiente de red, por lo que el servidor y el cliente no tienen que estar ubicados en la misma computadora.

Las distribuciones de Linux incluyen el sistema XFree86, que es un derivado del original X-Windows, específicamente adaptado para computadoras de la plataforma x86. El interfáz gráfico ya viene preconfigurado para trabajar de una forma "workstation" o "personal computer", o sea, su configuración se orienta en lo costumbrado de MS-Windows: una persona, una computadora. Estas preconfiguraciones son normalmente sencillos en su configuración, al menos en cuanto al Hardware, ya que hay una vasta cantidad de Chipsets de Video diferentes y XFree86 no desarrolla(ba) facilidades de instalar un driver (gestor) correcto. La adaptación de X-Windows a una tarjeta gráfica específica entonces puede resultar una tarea tediosa.

Quien está interesado solamente en una configuraciones Workstation estándar pero encuentra dificultades con XFree86 puede pasar directamente al capítulo de la configuración de Hardware (Tarjetas gráficas).

En los próximos subcapítulos se explicará la arquitectura del sistema X-Windows en más detalle para dar el fundamento a la configuración más avanzada de los diferentes componentes que integran el sistema operativo, la red, el interfáz gráfico y las aplicaciones en un sistema operativo Linux.

X-Windows fue percebido en el MIT - Massachusets Institute of Technologie, donde posteriormente los derechos fueron trasladados a X-Consortium Inc. y finalmente a la fundación de Software abierto: Open Software Foundation[1].

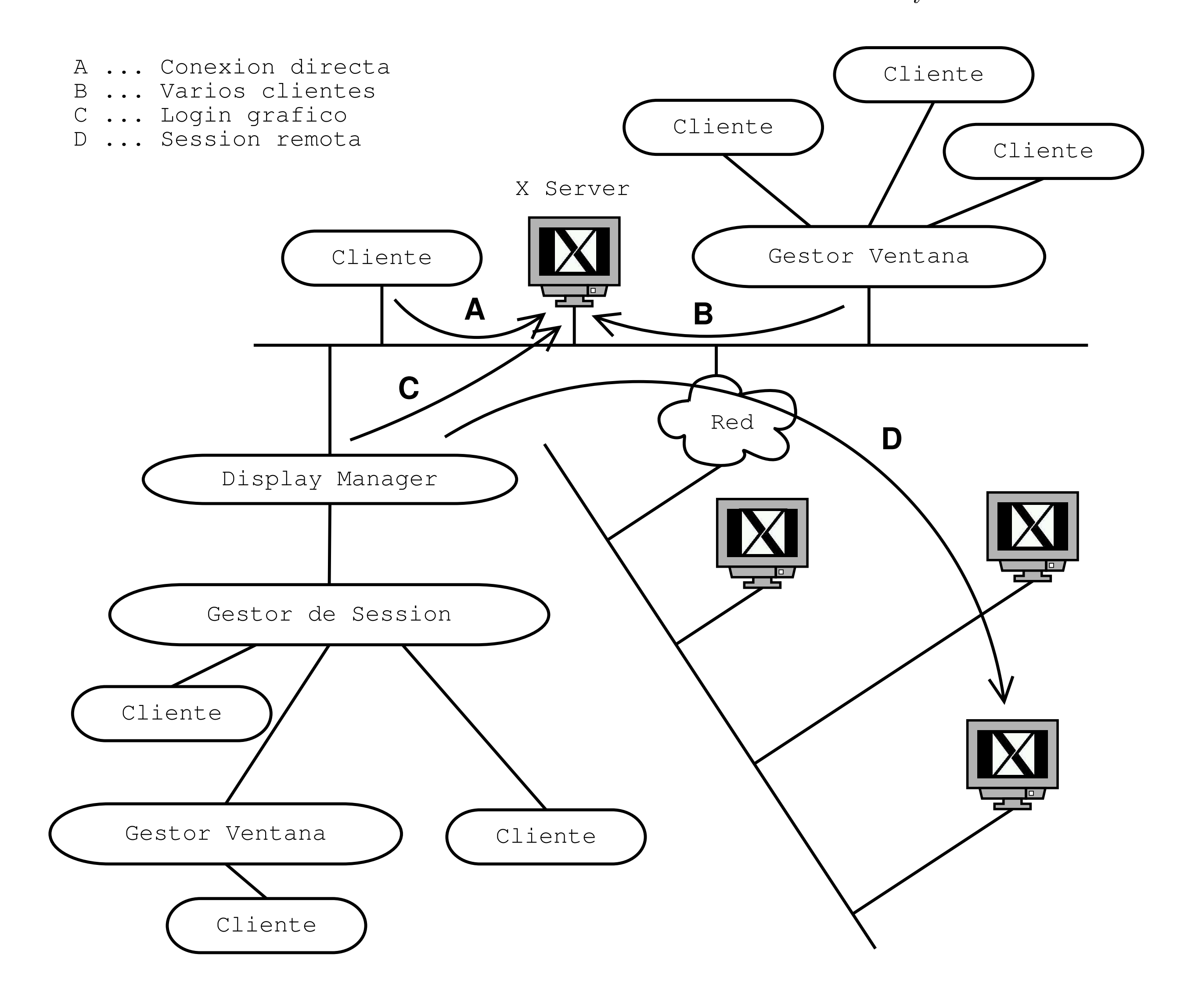
## 7.1. Servidor y Clientes

En el capítulo 5.2 se hace mención de la estructura cliente - servidor del sistema X-Windows. 'X' es el programa servidor, que "captura" los recursos pantalla, teclado, dispositivo puntero (y eventualmente otros) a la hora de arrancar, y permite a los programas de aplicación solamente el acceso a través de un protocolo específico con él. En cambió provee funciones avanzados para la graficación que pueden ser usados por los programas.

Para que un programa (cliente) pueda acceder a las funciones gráficas, tiene que *establecer una conexión* con el servidor X. Con un servidor pueden conectarse zero, uno o varios clientes; en efecto normalmente son una gran cantidad, ya que cada ventana representa un cliente.

La conexión puede realizarse de varias formas diferentes.

[1]No hay que confundir Software abierto con Software Libre. Ambos tienen percepciones muy diferentes y hubo mucho disputo sobre la liberación de X-Windows al público.




### 7.1.1. xinit, xstart.

Se puede utilizar solamente la “interior” parte de este esquema y correr un Servidor X. Para esto existen dos script: xinit y xstart. El segundo utiliza el primero para lanzar un servidor X local y algunos programas clientes, generalmente solo un gestor de ventanas o eventualmente de sesión. xstart provee un ambiente mas “seguro” y elaborado. La configuración de este modo de trabajo se efectua a través de los archivos xinitrc (global) y .xinitrc (individual por usuario).

Finalmente es posible correr solo una aplicación gráfica sin uso de un manejador de ventanas.

El comando para lanzar un servidor X es comúnmente “X”, aunque en el sistema Linux (i386) normalmente es un enlace hacia un ejecutable diferente, p.ej. XFree_SVGA, que es para tarjetas SuperVGA. “X” No se lanza manualmente, sino se utiliza los comandos xinit, xstart o un gestor de pantallas para ello. Al lanzar “X” de la línea de comandos se

observa simplemente una pantalla característica gris, en la cual se puede mover un cursor en forma de “X”.

### 7.1.2. Gestor de Pantallas - xdm

xdm es el “X Display Manager”, un programa que maneja conexiones entre clientes de X-Windows y Servidores a través de la red con un protocolo tcp/ip llamado xdmcp - “xdm - control protocol”. xdm solamente “negocia” la conexión entre un servidor X y una “sessión”, que es un programa que provee al/a la usuario/a con facilidades (gráficas) de lanzar programas de aplicaciones.

Vemos la parte de la conexión entre el servidor X y el xdm.

En la forma más sencilla, el servidor X corre en la misma máquina como el xdm. Normalmente entonces xdm lanza el o los servidores X según las indicaciones en el archivo /etc/X11/xdm/Xservers, cada línea describe un comando para lanzar un servidor X. La comunicación entre los dos programas se establece a traves de un socket unix, o sea mediante el núcleo del sistema y es muy eficiente. Varias tarjetas de video, pueden ser manejadas por un xdm, lo que se puede usar para efectos de pantalla grande, o p.ej. en programas avanzadas CAD donde una pantalla sirve para ingresar datos y comandos y otra para visualización del producto.

La segunda opción de conexión de un X server con su sessión es solicitar directamente una conexión con un server a través de la opción de la línea de commando -query. Ej. “X -query toa.magma.com.ni”. Si en la computadora toa.magma.com.ni corre un xdm y accepta la conexión desde la computadora solicitante entonces se establece la conexión entre ellos a través de la red. El servidor X tiene que lanzarse entonces de otra manera, p.e. a través del archivo /etc/inittab. Lógicamente pueden residir los dos programas en la misma computadora, es decir, no se lanza el servidor X desde el archivo /etc/X11/xdm/Xserver con el mismo xdm, sino con otro mecanismo, pero se deja conectar el servidor X con la computadora en la que está corriendo. El beneficio de este modelo es, que pueden ahorrarse recursos del sistema al no lanzar el servidor X cuando no esté en uso.

La tercera opción es, que un servidor X solicita una conexión indirecta (X -indirect toa.magma.com.ni). De esta manera el xdm de contacto retransmite la solicitud de negociación a todos los xdm en la red, el servidor X por lo tanto se hace “público” en todas las computadoras conectadas (con xdm corriendo). En esta forma, se presente inicialmente un selector de hosts (“chooser”) en la pantalla de la computadora, en la cual se visualizan todas las comutadoras en la red, que permiten al servidor X establecer una conexión y por lo tanto correr programas. Al seleccionar una computadora se re-establece una conexión entre el xdm de aquella computadora y el servidor X (la pantalla gráfica), y realiza una sessión entre estas dos computadoras. También en este caso eventualemente se corre la sessión y el servidor en la misma computadora.

Vemos la parte de los programas clientes.

Estos solamente se pueden lanzar desde dentro de una “session”. Para iniciar una sessión se requiere de un proceso de autentificación del usuario, en parte por la comprobación de derecho al acceso al recurso (del X server), pero en otra parte para preparar el ambiente

de trabajo para el usuario, quiere decir las configuraciones individuales de todos los programas y aplicaciones.

Veamos los diferentes escencarios del punto de vista del/de la usuario/a que está sentado frente a un terminal en el cual corre un X server. Si el servidor usa conexión:

**Indirecta:** xdm presenta una lista de todas las computadoras que ofrecen negociación mediante xdm a través del "chooser". El usuario selecciona una computadora con la cual quiere trabajar. Después pasa a la misma situación como en la conexión:

**Directo/local** El xdm presenta una ventana de autentificación (login-widget) en la cual el usuario puede introducir nombre y clave. Tomese en cuenta, que estos textos pueden transmitirse a través de la red y de esta manera exponerse a un "intruso":

Después de la autentificación (exitosa) se lanzan programas definidos en /etc/X11/Xsession, que normalmente proveen un gestor de sessión, un gestor de ventanas y aplicaciones iniciales según configuración individual de cada usuario. Todos estos programas son clientes conectados con el servidor X a través de uno (local/directo) o dos (indirecto) servidores xdm. Cuando el último cliente hay terminado, lo que pasa normalmente al "cerrar" la sessión (el gestor de sessión) el servidor X se desconecta del servidor xdm y vuelve a conectarse con la solicitud directa o indirecta, o en caso de un servidor local va a ser lanzado de nuevo desde xdm.

A parte del tradicional "xdm" existen otras re-implementaciones del protocolo xdm, en específico: wdm - del ambiente Windowmaker, kdm - del Desktop KDE y gdm del Desktop Gnome. El último hasta ahora no provee la funcionalidad para conexiones indirectas.

**Gestor de sessión:** tiene la función, de proveer el "ambiente" de la sessión. Este programa graba por ejemplo la posición de las ventanas, registra cuales programas están "abiertos" a la hora de terminar una sessión, y los vuelve a lanzar y poner en la misma posición la próxima vez cuando el/la usuario/a abre una sessión.

**Gestor de ventanas:** Es el programa, que durante la sessión permite lanzar y cerrar programas (mediante su visualización en ventanas) y reorganizar su aspecto visual (iconificar, maximizar, redimensionar, etc.). Hay administradores que incorporan otros elementos de mando como iconos, barras de tareas, menúes desplegables (p.ej. qvwm), en otros sistemas estos elementos son programas independientes (Gnome Desktop: sawfish -Window Manager, panel - Barras de tareas y menúes, gmc - "Explorador").

Para crear una analogía al trabajo en modo texto (consola) de unix se puede decir que xdm es como el programa logín, que permite autentificarse al sistema, la sessión corresponde a las preconfiguraciones del interpretador de comandos de entrada (login shell, p.ej.: bash: /etc/profile, ~/.bash_profile, ~/.bashrc, ~/.profile), y el gestor de ventanas corresponde al interpretador de comandos, al shell - es un shell gráfico.

Figura 7.1.: Descomposición matricial ocho por ocho

## 7.2. Configuración

En este capítulo resumimos primero algunos conceptos técnicos esenciales de los ambientes gráficos, para después mencionar varios programas que permiten crear la configuración de un servidor X para una determinada combinación de tarjeta de video, monitor y computadora.

Actualmente existen distribuciones Linux con la versión 3.3.6 de X-Windows, pero también con la versión 4.0. Las dos versiones tienen diferencias sustanciales en cuanto a la configuración: para la versión 3.3.6 se requiere mucho más conocimientos para la configuración correcta, que para la versión más nueva. Donde es posible se trataría ambas versiones, en cuanto hay diferencias relevantes se hará mención específica acerca de la versión en el texto.

### 7.2.1. Categorías de tarjetas gráficas

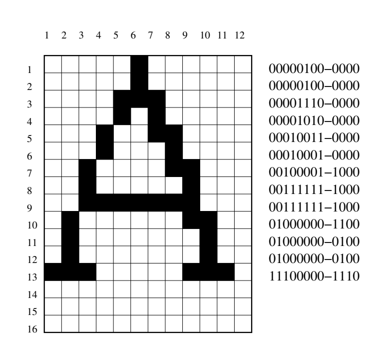
En los sistemas compatibles con el IBM-PC encontramos una ámplia gama de diferentes tarjetas gráficas. Su función es, crear una imagen visual en una pantalla de tubo de cátodos (CRT/Monitor), una pantalla de cristál líquido (LCD), o equivalente.

La comunicación entre el contenido visual y la computadora se realiza a través de un área secuencial de memoria RAM, en la cual la computadora escribe información binaria, que va siendo convertido en una imagen bidimensional en el dispositivo de visualización.

Distinguimos dos tipos principales de conversión binaria/visual:

**Character mapped** donde en cada celda de memoria se escribe el código de cáracter: a, Z, !, etc. La memoria video se divide en secuencias de longitud fija (comúnmente 80), que corresponden a las líneas en la pantalla (comúnmente 24 o 25) .La tarjeta gráfica lee los caracteres secuentialmente línea por línea y los dibuja en la pantalla de la izquierda a la derecha, desde arriba hacia abajo. La conversión de la letra "a" a su figura gráfica se realiza dentro de la tarjeta gráfica, muchas mediante un "generador de caracteres", que muchas veces tiene programado fijamente en memoria ROM la aparencia de cada letra (en descomposición matrizial). En algunas tarjetas se pueden cambiar algunas letras o todo un juego completo (font) de caracteres.

Figura 7.2.: Descomposición matricial 12x16




Cuadro 7.1.: Descomposición matricial

| |
|---|
| En las figuras 7.1 y 7.2 se presenta la descomposición matricial de la letra 'A' con dos diferentes mátrizes. Se supone una representación monográfica donde un bit 0 en la memoria corresponde a un punto "oscuro" en la pantalla y un bit 1 corresponde a un punto "iluminado". Mientras en la primera descomposición se utilizan solamente $8\,Bits \times 8 = 8\,Bytes$ de memoria, queda muy restringida la reproducción gráfica.<br>En la segunda descomposición se utiliza una mátrix (un 'raster') de 12 por 16. Se consumen $12\,Bits \times 16 = 1{,}5\,Bytes \times 16 = 24\,Bytes$ |

**Bit mapped** donde cada celda de memoria representa un "pixel" - un punto gráfico en la pantalla. La pantalla está organizada como matríz en líneas (p.ej. 600), las cuales están dividas en puntos (p.ej 800). La tarjeta gráfica lee secuencialmente cada celda de memoria y exita la luminosidad y el color del pixel correspondente de la pantalla según su contenido. Si se pretende mostrar una letra en una determinada posición en la pantalla hay que realizar la descomposición matrizial según el "font", su tamaño en la pantalla y el color deseado, y escribir el contenido de las celdas de memoria correspondiente, sin apoyo de la tarjeta gráfica.

En la actualidad ya no se trabaja con tarjetas "character mapped", sino las tarjeteas gráficas proveen varios "modos", entre ellos modos "texto" donde simulan una tarjeta character mapped con diferentes cantidades de columnas y líneas y tipos de letras, y modos "gráficos" con diferentes cantidades de líneas y columnas de "pixels".

Existe una grán variedad de tipos de tarjetas para la computadora PC:

**Monocromático** el adaptador gráfico del primero PC, que solo provee modo texto en un color. La luminosidad de una letra puede ser nula, baja, normal o alta (highlight) - negrilla. Requiere un monitor monocromático con un conector de nueve pines.

**Hercules** que es un adaptador monocromático mejorado. Trabaja con el mismo monitor monográfico.

**CGA** adaptador a colores, que provee modos gráficos y textos de baja resolución (vea más adelante) y hasta 16 diferentes colores. Requiere un monitor a color específico, también con un adaptador de nueve pines.

Estos adaptadores hoy día ya no se aplican.

**VGA** es un adaptador bit mapped que provee varios modos gráficos y de texto, y hasta 256 colores en cada pixel. En su forma estándard tiene una resolución de 640x480 puntos. Aunque diferentes empresas producen diferentes circuitos integrados para realizar la conversión visual de la información a la pantalla, hay un estándar para su programación. En efecto, hoy en día se puede confiar, que cualquier computadora compatible puede trabajar en forma VGA básico.

A partir de la tarjeta VGA empiezan a surgir tarjetas gráficas con una diversificación de capacidades de colores y resoluciones, se hace accesible el ambiente del diseño gráfico en las computadoras personales.

Se requiere de un monitor específico para este tipo de tarjetas - el monitor VGA. Existen con conectores de nueve y de 15 pines.

**SVGA** o Super-VGA, que es una extensión del modo VGA a una resolución de 800 por 600 puntos en la pantalla. En la actualidad solamente se producen adaptadores gráficos con capacidades SVGA y muy superior. Pueden usar monitores VGA o SVGA.

Con el aumento de la capacidad en cuanto a resolución matrizial y de color se van aumentando dos parametros fundamentales en la comunicación entre la computadora y la tarjeta gráfica: la cantidad de datos a transmitir por cada imagen en la pantalla y el tamaño de la memoria video necesario para almacenar la imagen.

Las primeras tarjetas gráficas usaron el bus (lento) de entrada/salida de la CPU intel mediante el conector ISA de ocho bits (transferencia de un caracter por transacción), a continuación expandiéndose al bus EISA de 16 bits, y finalmente el bus PCI de 32 bits y de mucho mayor velocidad (permitiendonos saltar otros buses que no tienen mucha importancia en la actualidad).

La memoria video siempre fue incorpordado en la tarjeta gráfica. Ya que requiere de una memoria de alta velocidad y por lo tanto de alto costo muchas veces la tarjeta puede provisionarse con menos o más chips de memoria de video, según necesidad y posibilidades económicos.

En la actualidad está en uso común el sistema AGP, donde la CPU comparte memoria principal con la tarjeta gráfica. Esto tiene la ventaja (mercadotécnica) de reducir sustancialmente el costo del adaptador gráfico, a costo de los recursos disponibles para los programas. El monto de memoria de video reservado para el adaptador gráfico se determina en la configuración CMOS de la tarjeta madre.

Una ventaja de este sistema se da en cuanto a la velocidad del acceso a la memoria de video, ya que solo se trata de copiar información a un lugar específico de la misma memoria principal. Esta ventaja se paga con una mayor carga de la CPU para el procesamiento de la información gráfica.

Esta forma de acceso compartido a la memora video también se denomina "framebuffer", donde "frame" es un marco - una imagen en la pantalla.

Para computadoras diferentes al PC se utiliza como concepto genérico de acceso, ya que en ellos no se puede contar con los mismos gestores de tarjetas VGA o SVGA.

El "framebuffer" es un modelo de abstracción de la memoria de video que ha hecho posible portar los programas gráficos (en especial el X-Windows en su forma de XFree86) en cuestión de muy poco tiempo a otras plataformas, como p.ej. el PowerPC de Macintosh.

### Monitores

En las tarjetas VGA hubo solo pocos modos gráficos las cuales exigen a los monitores CRT cumplir con ciertas características, específicamente con frecuencias de sincronización horizonta y vertical fijas.

Con el aumento de la diversidad de los modos gráficos aumentó la cantidad de frecuencias y surgieron los monitores "multisync", que permiten un rango variado de frecuencia para la sintonización entre monitor y tarjeta de video y facilitan sustancialmente la configuración. Notese, que hay monitores de frecuencia fija que pueden dañarse electrícamente si la tarjeta de video no produce las frecuencias específicas de sincronización para el modelo de monitor respectivo.

Para uso normal de oficina se considera suficiente el tamaño de "14 pulgadas" en un monitor. Esta medida indica la longitud de la diagonal del tubo de imágenes. Esto corresponde a un ancho de la pantalla de aproximadamente diez pulgadas. Tomando en cuenta que algunos modelos dibujan dentro de un marco de una pulgada desde la orilla del tubo de imágen quedan monitores de un ancho visible de ocho pulgadas.

Para diseño gráfico o trabajo extenso de programación etc. se utilizan monitores de 20 pulgadas. Estos ya requieren mínimamente un modo gráfico con una resolución mínima de 800 por 600 puntos para proveer una imagen clara y por lo tanto no es recomendable emplearlos con tarjetas gráficas inferiores a SVGA. Su costo es además muy elevado, y para trabajos intermedios existen monitores de 17 pulgadas.

Los tamaños recomendables y accesibles de pantallas son entonces: 15, 17 y 20 pulgadas.

### Resolución

El término resolución gráfica en adaptadores de video se refiere a la cantidad de puntos (pixel) horizontales en una línea, y de puntos (pixel) verticales en una columna, lo que corresponde a la cantidad de líneas (de pixels) visibles en la pantalla.

Generalmente se utiliza una relación de tres a cuatro entre la resolución vertical y horizontal, que es una aproximación a la proporción ideal de un rectángulo. Las resoluciones comunes son:

**680x480** buena visibilidad en pantallas hasta máximamente 17 pulgadas.

**800x600**

**1024x768** pantallas de mínimamente 14 pulgadas.

Si utilizamos una pantalla de 14 pulgadas con un ancho visible de nueve pulgadas y una resolución de 680x480 pixel, obtenemos aproximadamente 75 pixel (puntos) en cada pulgada. Esta "densidad" de puntos por área (es la misma en sentido vertical como horizontal) se mide en dpi - dot per inch o puntos por pulgada (ppp).

Dado la capacidad del ojo para resolver objetos hasta un cierto tamaño se considera, que 75 dpi resultan en imagenes o contornos "suaves" o "continuos", aunque en realidad están compuestos por una matríz rectangular de puntos individuales con diferente color y luminosidad. Si reducimos la densidad de puntos debajo de 75 dpi el ojo libre distingue facilmente los puntos individuales y comienza a perder el concepto de la imagen. Notese que en la técnica de impresión el término un punto ("one point") también se utiliza como la medida de 1/72 pulgadas, lo que corresponde a una densidad de 72 dpi; por este término enraizado utilizamos en monitores preferiblemente el término de "pixel" para designar un punto visible, notesé que un pixel puede tener un tamaño variable en diferentes monitores o resoluciones, mientras "one point" siempre tiene el mismo diámetro de 1/72 pulgadas.

Si tratamos de dibujar 80 caracteres en una línea en una pantalla de resolución 640x480 obtenemos que podemos asignarle ocho puntos consecutivos de ancho a cada letra; dejando un punto de distanciamiento nos quedan siete puntos, lo que es suficiente para una descomposición matrizial legible de todo el abecedario.

Con esto conseguimos los límites inferiores de desempeño gráfico: Si utilizamos un monitor con un ancho visible de nueve pulgadas (diagonal de 14") a una resolución de 680x480 pixel, podemos visualizar un objeto gráfico en su *tamaño real* siempre y cuando usamos un ráster de 75 dpi para la descomposición matrizial.

Para obtener una mayor calidad de resolución óptica podemos usar por ejemplo un monitor de 17 pulgadas, con un ancho visible de 14 pulgadas, para obtener una visibilidad a tamaño real con una resolución de 1024x768 y una densidad de 100 dpi.

En este modo (1024x768) podemos dibujar líneas de texto con 128 caracteres por línea, o por otro lado usar hasta 12 puntos para la descomposición matricial de una letra, asumiendo una línea de 80 caracteres.

### Profundidad de color

En cada pixel de una pantalla se puede regular intensidad y en monitores a colores también el color. En la mayoría de las pantallas los colores se producen mediante superposición de los colores básicos aditivos azul, rojo y verde. En realidad un pixel se compone entonces de tres pixels, cada uno en un color báscio, y se regula la luminosidad mediante la composición de las luminosidades de los tres pixels de color.

Si utilizamos por ejemplo un número de 0 a 15 para regular la luminosidad de un pixel de 0 a 100 %, requerimos de $3 \times 4\,Bits = 12\,Bits = 1{,}5Bytes$ para representar luminosidad total y color de un pixel. En este caso decimos que la profundidad de color es de 12 Bit. La mezcla de colores con todas las combinaciones posibles de los tres pixels de color básico puede producir $16 \cdot 16 \cdot 16 = 4096$ colores diferentes[2].

[2] Esto requiere, que también la pantalla puede reproducir fielmente todos los niveles de luminosidad en todos los colores.

Una tarjeta moderna puede tener para cada color un byte, lo que corresponde a una profundidad de colores de 24 Bits.

Usando esta terminología podemos también clasificar pantallas monocromáticas.

| | Profundidad/Bits | Nº de colores | |
|---|---|---|---|
| Monocromático | 1 | 2 | Blanco/Negro |
| Hercules | 2 | | 4 niveles de intensidad |
| CGA | 4 | 16 | |
| VGA | 8 | 256 | Usa paleta de colores. |
| | 15/16 | 32768 | $3 \times 5$ Bits |
| True Color | 24/32 | 1.7 Millones | 3 Bytes, el cuarto byte a veces no se usa por razones de eficiencia. |

El concepto de la paleta de colores explicamos mediante un ejemplo: para cada color básico se utilizará una rango de luminosidad de 5 Bits. La paleta es una lista de números de 0 a 255, a los cuales se les asigna una combinación de luminosidades que utiliza 15 Bits o dos Bytes respectivamente.

De esta manera se pueden crear tonos intermedios, que no se pudieran realizar utilizando 8 Bits. Con ocho bits solo se pueden utilizar unidades de dos Bits para cada color básico resultando en cuatro niveles de luminosidad, y correspondiente a una profundidad de colores de 6 Bits, o 64 colores diferentes. Con la paleta se pueden crear cuatro veces más colores diferentes.

X-Windows tiene la posibilidad de usar *paletas de colores privadas* o compartidas. La paleta privada permite a cada aplicación utilizar en todas sus ventanas 256 diferentes colores. En el momento que el cursor sale del área de la ventana de la aplicación el servidor X cambia la paleta de colores, por lo que la ventana cambia bruscamente de color, lo que puede ser iritante. Muchas aplicaciones proveen una opción para utilizar una paleta compartida (shared colormap), que entonces no refleja fielmente algunos colores, pero evita este efecto (colormap switching).

### Video Chipset

El estándar VGA/SVGA deja cierta libertad para extensiones, que han fomentado que diferentes producentes de circuitos electrónicos (Chips) han creado dispositivos con una variedad de características, que pueden ser determinado por un lado por mejorar la calidad de la imagen, la velocidad de la visualización o por el otro lado por una reducción en los costos de diseño y producción de las tarjetas de video. Algunas empresas se han destacados y creado familias de dispositivos, mejorando o ampliando paulatinamente sus diseños. Una série específica de dispositivos que en su conjunto integran la función de proveer a la CPU un interfáz VGA o SVGA y producen las señales de video para la pantalla se llama "Chipset", que significa nada más que "conjunto de chips".

Un resultado de esta diversificación es, que a pesar de ser un estándar, VGA y SVGA no permiten la creación de un gestor de hardware genérico, sino cada Chipset necesita

su propio gestor para utilizar las capacidades específicas o ampliadas. Aunque algunas "familias" de Chipset pueden utilizar un solo gestor, esto no es válido para todos.

En la versión 3.3.6 del servidor X-Windows XFree86 el gestor de la tarjeta de video es compilado fijamente (estático) con el resto de las rutinas. Por lo tanto existen varios diferentes "servidores" o más preciso versiones del programa X, cuales incluyen los gestores para una o más familias específica de Chipsets.

Casi todas las tarjetas SVGA pueden ser usados con el servidor XFree86_SVGA, con excepción de algunos servidores "accelerados".

Tarjetas VGA usan el servidor XFree86_VGA y tarjetas monocromáticas el servidor XFree86_Mono. Vale mencionar, que una tarjeta SVGA también puede usarse en modo monocromático o VGA con el servidor "inferior" respectivo, así como la tarjeta VGA puede trabajar en modo monocromático con XFree86_Mono.

Usando una tarjeta en un modo inferior a su capacidad no es aberado, sino resulta en una reducción de memoria occupada, un aumento drástico de velocidad, o permite usar una resolución muy superior. Para ciertas aplicaciones donde la profundidad de colores no tiene importancia esto puede ser de ventaja.

La versión 4.0 del XFree86 se destaca por una separación de los gestores de la tarjeta del resto del código, lo que permite una mayor flexibilidad, y para las y los usuarias/os una instalación y configuració sustancialmente simplificada. Sin embargo no existen gestores para todas las tarjetas gráficas existentes en la versión 3.3.6, por lo que hay que tener cautela al adquirir una tarjeta gráfica nueva, para que sea compatible con la versión 4.0, o por otro lado hay que tomar en cuenta que puede ser necesario la instalación de la versión "vieja" de X-Windows, especialmente en tarjetas "viejas".

### Servidor de tipos de letras

La tarea de presentar texto (font rendering) en el ámbiente gráfico casi en todos los sistemas es asumido por el programa servidor con funciones dedicadas a la visualización de texto. En modo texto la tarea es más fácil, ya que se utilizan una matríz de caracteres con ancho fijo por cada celda. El modo gráfico permite el uso de tipos de letras con ancho variable por cada letra, lo que produce un flujo del texto más legible para el ojo humano.

Los problemas técnicos que se presentan en este proceso es la determinación del espacio que ocupará un cierto texto, y la conversión del código de caractér en su correspondiente símbolo gráfico, en dependencia del tamaño requerido (rendering).

X-windows puede leer archivos de fuentes y realizar la visualización, pero existe también un protocolo "font-server" - servidor de tipos de letras (xfs), que le puede asignar a un programa eventualmente remoto asumir esta tarea. Un objetivo de esto puede ser el de desligar a una computadora de la tarea de calcular la forma visualizada de un texto, y de esta forma ahorrar recursos del procesador, tomando en cuenta que una computadora que alberga un programa font-server puede trazar los fuentes para muchas otras en la red. Otro objetivo puede ser el de solo mantener una base de datos de fuentes en una red. Cada archivo de fuente que es agregado a la computadora con el font-server automáticamente es disponible para todas las computadoras en la red. Esto ahora espacio en los discos duros, evita duplicidad y más que todo requiere menos trabajo de configuración y

mantenimiento, ya que solo tiene que realizarse en un solo punto.

Entre los fuentes se reconocen, sin pretender de abarcar todos, cuatro tipos importantes en cuanto a su apariencia:

**Roman** - fuentes con serifes - que son las colitas al final de las líneas réctas con que son formados las letras. Esta letra se utiliza por su gran diferenciación en textos largos como libros. Un ejemplo importante son las diferentes fuentes "Times". En MS-Windows Times New Roman, en Unix "Times" y en Tex/LaTeX "Computer Modern Times".

**Sans Serif** fuentes simples sin serifes, "Arial" en MS-Windows, Helvetica en StarOffice

`Typewriter` fuentes fijos, o fuentes de "máquinas", donde cada letra tiene el mismo ancho. En MS-Windows: "Courier".

En cuanto a su representación como archivo de tipo de letra hay varios formatos. En primer lugar se puede diferenciar entre

**Bitmapped fonts** donde el archivo contiene la descomposición matrizial que hemos visto en el capítulo sobre las tarjetas de video. Estas fuentes no se reproducen bien cuando son ampliados o reducidos de tamaño y además requieren mucho espacio para su almacenamiento.

**Outline fonts** donde el archivo contiene parametros para una reconstrucción de la letra individual mediante algún algorítmo, p.ej. Postscript Type 1, o TrueType de Microsoft. La descomposición matrizial se realiza hasta el momento cuando es requerido una letra, y se realiza con los parámetros de densidad (dpi) y tamaño justo, por lo que estas letras tiene mucho mejor presentabilidad en la pantalla, u otro medio de reproducción.

Se puede diferenciar la información sobre la forma de la fuenta, y la "métrica". Para determinar el espacio que va a occupar un cierto texto solo se requiere las dimensiónes exteriores de cada letra usada y no su forma. En algunas representaciones de tipos de letras (fonts) la información métrica se almacena en un archivo separado de la forma de las letras.

Mientras los archivos de tipos de letras TrueType anteriormente solo podían ser usados en los sistemas operativos de Microsoft ahora igual el servidor X versión 4.0 soporta su formato, y también existe un programa servidor de fuentes `xfs-tt` que puede rasterizar fuentes estándard de Unix y de X-Windows, como las fuentes TrueType. Esto significa, que las aplicaciónes Linux pueden visualizar e imprimir documentos con métrica y forma idéntica a las aplicaciones de MS-Windows. Los tipos de letras básicos de Microsoft Windows son disponible gratuitamente en Internet, gracias a una política colaborativa de esta empresa; su manejo difiere en diferentes distribuciones. En la distribución Debian hay un soporte directo y una utilidad para descargar y actualizarlas directamente.

### 7.2.2. Xconfigurator

La distribución Linux de RedHat 7.1 todavía está usando la versión 3.3.6 de X-Windows, posiblemente por la mejor compatibilidad con la gran variedad de Chipsets de Video.

RedHat provee un programa excelente de configuración para X-Windows, llamado Xconfigurator, que es también parte del programa "setup", desde donde preferiblemente se llama.

El atractivo de la configuración X en RedHat es una base de datos de tarjetas de video y monitores con sus datos de configuración predifinidos, que permite una configuración mediante el nombre.

Se entiende que la configuración de X-Windows no puede realizarse en modo gráfico, pero RedHat utiliza un sistema semi-gráfico (en modo texto) para sus programas de configuración. Xconfigurator solicita al usuario los datos necesarios mediante menúes de selección, y crea desde allí un archivo de configuración para X-Windows, que normalmente es muy acertado.

Vale notar, que RedHat utiliza un enlaze simbólico /dev/mouse para definir el dispositivo donde está conectado el ratón. La configuración del sistema configura a gpm (el gestor del ratón en modo texto) y a X-Windows para que utilizan este enlace, y para que cada uno, gpm y X, utilizen su propio gestor correspondiente al tipo de ratón. Esto es diferente al concepto utilizado con la distribución Debian el cual se presentará a continuación.

La consecuencia de esta configuración es, que al cambiar el ratón posiblemente solo hay que cambiar el enlace simbólico y gpm y X reconocen autoáticamente el nuevo dispositivo, sin embargo hay que cambiar el gestor en ambos programas cuando se cambia el tipo de ratón también.

### 7.2.3. XF86Setup

Es un programa proveido por la misma distribución de XFree86. El concepto de este programa es, de lanzar un servidor X en modo VGA - estándar para tener un ambiente gráfico, desde el cual se selecciona todos los parámetros cómodamente.

Sin embargo, este programa en muchas situaciones no funciona. Si se dispone de una tarjeta gráfica no compatible con el servidor XFree86_VGA el programa no arranca, en caso que sí es compatible, muchas veces tarda en arrancar. Otro problema es, que no es fácil para XF86Setup adivinar dispositivo y tipo de ratón, por lo que muchas veces no son utilizables y los menús se hacen difícil de manejar, en especial la selección de la tarjeta de video en la lista, que es extremadamente larga.

XFree86 en la versión 3.3.6 provee una grán base de datos de tarjetas de video, y selecciona de ellos el programa servidor X correspondiente. Sin embargo muchas veces no funciona bien el archivo de configuración creado, y el servidor X no logra sincronizar la pantalla.

De mucha ayuda es el archivo de Monitores, donde hay las frecuencias y parámetros funcionales para una inmensa variedad de Monitores CRT, que pueden consultarse e

insertarse manualmente en el archivo de configuración de X-Windows.

### 7.2.4. xf86config

Es la utilidad genérica para la configuracion de XFree86. Es un programa solo de texto, en el cual se contesta una variedad de preguntas, que son utilizadas para crear un archivo de configuración. Aunque sufre de los mismos defectos como XF86Setup en cuanto a la producción de archivos de configuración no funcionales.

Sin embargo lo considero como la herramienta básica para la configuración, que en muchos casos es la única forma de generar un archivo de configuración básica, que puede modificarse posteriormente. Por eso es válido utilizar este prógrama y tratar de conocer bien su conocimiento, además de que tiene texto de ayuda bien elaborado, que da al/a la usuario/a conocimiento importante sobre los aspectos de configuración de X-Windows.

### 7.2.5. dexconf

Es un programa de la distribución Debian de Linux, que normalmente no se utiliza manualmente, sino forma parte de la configuración automática. Se puede usar para crear una configuración base, que muchas veces ya es satisfactoria, o puede ser modificado facilmente para adaptarse a las necesidades individuales. dexconf existe desde la versión Woody (3.0) de la distribución Debian, donde se aplica ya el servidor XFree86 versión 4.0. En esta distribución es además muy bueno el programa de configuración, que puede volver a llamarse en cualquier momento con la línea de comando:

```
dpkg-reconfigure xserver-xfree86
```

El único defecto es, que el programa no toma en cuenta el tipo de teclado utilizado en la computadora, genera una entrada para el teclado inglés, y peór: aunque se corrige esta entrada manualmente se encontrará sobreescrito por la versión "anglocéntrica" en cada actualización de xserver-xfree86.

En Debian pueden instalarse la versión 3.3.6 y la versión 4.0 de XFree86 en paralelo. En la versión 2.2. (Potato) solo existe la versión 3.3.6, donde la configuración preferiblemente se realiza con las herramientas básicas descritas en las secciones anteriores.

### 7.2.6. XF86Config

Es el archivo de configuración para el X-server, que normalmente se encuentra en el directorio /etc/X11. Este archivo a propósito permite redundancia, quiere decir que pueden incluirse configuraciones para varios monitores, tarjetas gráficas o configuración de los dispositivos de ingreso - ratón (pointer device) y teclado.

De esta forma se pueden crear archivos de configuración, que son válidos para varias configuración de computadoras, lo que puede ser importante para administradores de sitios con muchas computadoras como centros de cómputo, universidades, etc. A la hora de leer el archivo de configuración el X-server valida o descarta las líneas individuales para formar la configuración correspondiente a sus capabilidades. El archivo XF86Config

utiliza un "lenguaje" de configuración simple pero bien estructurado, mediante la cual se divide el archivo en diferentes secciones: "Section's". Dentro de cada sección se configura algún aspecto del servidor. Hay secciones globales, que albergan opciones globales de configuración de los servidores X de la computadora local, y secciones que pueden ser individuales, por lo que se declaran etiquetas, llamadas "Identifier", mediante en las últimas secciones se determina, cual parte del archivo de configuración se toma en cuenta en un servidor específico.

Un elemento importante para el éxito de la configuración son las líneas de modos gráficos, los "Modelines", y la configuración de frecuencias verticales y horizontales del monitor.

Los Modelines efectivamente programan el Chipset y determinan la resolución a utilizar. Pueden utilizarse varias en una sola sesión X y cambiarse de una resolución a otra mediante una combinación especial de teclas. Para cada combinación de monitor, tarjeta de video y resolución es preciso una correcta configuración para obtener una imagen en el monitor. En la configuración estándar se insertan muchas Modelines con la misma resolución, pero con frecuencias de sincronización ascendentes en XF86Config, o sea, las líneas para tarjetas y monitores inferiores vienen primero. A la hora de leer el archivo el X-server descarta todas las líneas que no pueden cumplir con las restricciones del monitor, y se queda la última y por lo tanto "mejor" línea de configuración para la resolución respectiva. Esta por su lado pueda que no funcione, cuando se especifico datos del monitor no correspondientes a su capacidad. En este caso se puede marcar la última línea (la usada sin éxito) como comentario y volver a probar arrancar el X-server de nuevo.

Un método de comprobar si el archivo de configuración es válido es, utilizar la línea de comando:

```
X -probeonly 2>xserver.txt
```

De esta forma, el X-server lee y ejecuta el archivo de configuración, pero sin cambiar al modo gráfico. Al terminar de ejecutarse, se puede analizar el archivo `xserver.txt` en el directorio actual, para determinar que opciones son usados por el servidor X y donde eventualmente habrá un problema. Notese, que hay problemas con X-Windows que no pueden dedectarse de esta forma, por ejemplo puede haber la ausencia de cierto tipos de caracteres en su sistema, lo cual solo se detecta hasta que se lance la sessión X "real".

### 7.2.7. XF86Config-4

La versión 4.0 utiliza un archivo de configuración con una estructura parecida a la de la versión 3.3.6, pero que no es compatible.

En especial hay cambios en cuanto a los dispositivos de entrada: teclado y ratón, en cuanto a la composición de los diferentes opciones para crear "screens" - configuraciones de una o más pantallas, y además hay que especificar cuál gestor de hardware utilizar, ya que no son parte interno del servidor.

Sin embargo, en total se ha facilitado la configuración sustancialmente y el servidor X realiza mucho trabajo de auto-configuración, tomando valores sensatos por defecto.

Para evitar problemas de competencia entre los dos versiones X, el servidor 4.0 trata de leer primero un archivo con nombre "`XF86Config-4`", y solo si no encuentra este utiliza el nombre "`XF86Config`" como patrón de búsqueda. Por lo tanto puede llamarse el archivo de las dos formas, pero se utiliza preferiblemente la primera, para permitir eventualmente correr un servidor anterior.

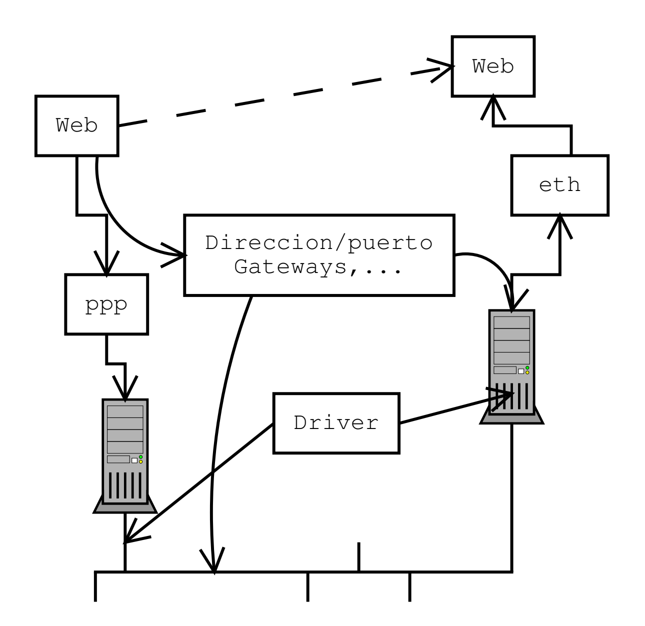
# 8. Red TCP/IP

El Internet hoy en día es una palabra que designa una red que físicamente abarca todo el planeta tierra, mas común aún se utiliza para describir una aplicación específica: www, o http. Pero en su percepción original es una colección de servicios y protocolos de red. TCP significa Transfer-Control-Protocol, e IP significa Internet Protocol. Si hablamos de redes Internet puede ser entonces la implementación mundial específica, pero también si en una oficina se conectan dos computadoras Linux via una red se trata de una red Internet. Para distinguir esta red pequeña de la red Internet mundial se le suele llamar Intranet a estas configuraciones, aunque el funcionamiento es idéntico.

La conexión de dos o mas computadoras siempre sigue los mismos mecanismos, los que se han identificados y normados por ejemplo en el modelo ISO-OSI de siete capas. Sin entrar en el detalle de este modelo nos adherimos al modelo de capas. Solamente contemplamos las capas de interés para la configuración del nucleo Linux. Desde "abajo" hacia "arriba" diferenciamos entre:

- Driver (gestor). Programa que controla el dispositivo físico de transmisión, tarjeta de red, modem, etc.
- Protocolo. Norma de intercambio de datos entre dos dispositivos. PPP, TCP/IP, PLIP, etc. programas que implementan esta norma bajo utilización de un Driver.
- Enrutamiento. El mecanismo que determina el camino que toman los datos a transmitir si existen varias conexiones (en Linux mínimamente dos si usamos cualquier conexión a una red), routing table.
- Servicios. Programas auxiliares utilizados en la administración de conexiones entre diferentes computadoras. DNS, ICMP, ARP, SMTP, etc.

El enrutamiento en realidad es parte de las capas inferiores. En la figura 8 se ilustra con la línea rayada la comunicación entre una computadora (izquierda) que utiliza el protocolo http para solicitar información de otra (derecha). Seguimos ahora el flujo de comunicación "real" desde la computadora izquierda a la computradora derecha. Se requiere de la seleción del destino (Dirección/puerto - información para el enrutamiento), los datos son empacados en un protocolo (ppp a la izquierda). Los driver del hardware, presuntamente en este caso para manejar un modem convierte los datos en señales analógicos que se transmiten a través de la línea telefónica. La "red" dibujado abajo puede contener multiples caminos para los datos, pero el enrutamiento asegura, que los datos llegan a la computadora específica indicada en la derecha, en nuestro caso presuntamente ya convertido en señales de una red local Ethernet. La tarjeta ethernet convierte estos




Ejemplo de una conexión en red

en paquetes de datos, el driver "eth" extrae de estos la información TCP/IP y la remite en "texto claro" a la aplicación destino.

Se puede apreciar, que todos los componentes/todas las capas tienen que concordar en su configuración para que pueda establecerse una comunicación. A continuación en este capítulo se discuta la configuración de cada "capa" y los posibles problemas emergentes.

## 8.1. Drivers

Los driver de hardware es la parte del sistema operativo que comunica el mundo exterior a través de dispositivos de entrada/salida con los programas. Cada tipo de dispositivos requiere un driver específico. Por suerte hay ciertas normas, que permiten ordenar los dispositivos en grupos y escribir driver's genéricos para un amplio rango de dispositivos reales.

Los dispositivos más comúnes para la conexión en red son:

**modem** establece conexión a través de una línea telefónica mediante modulación/demodulación de datos en señales de diferentes frecuencias. Normalmente un modem es conectado en un puerto en serie (/dev/ttyS*), o lo simula. WinModems (HSP) no pueden ser usados en Linux, por un lado por falta de información técnica, ya que es propietaria, por otro lado porque funcionan un concepto técnico que requiere limitar la funcionalidad de la CPU de la computadora, lo que no se considera conveniente en Linux. Los protocolos usados para la transmisión de datos son PPP (Point-to-Point-Protocol), y SLIP (Serial-Line-Internet-Protocoll) que es obsoleto.

**tarjeta“driver”,** los más conocidos son NE2000, DEC/Tulip, 3COM. El dispositivo es accesible en linux como /dev/eth*.

**puerto paralelo** Como una solución barata para conectar dos computadoras (point to point) se utiliza un cable especial que conecta en vez de una impresora otra computadora con el puerto paralelo. El interfaz es físicamente el del puerto paralelo, aunque hay variedades, el dispositivo es llamado /dev/plip*, ya que el puerto utiliza otra forma de control a que cuando se conecta una impresora.

El protocolo usado se llama PLIP (Parallel-Line-Internet-Protocoll). Existe el modo 0 y el modo 1 de transmisión. El modo 1 es más moderno y rápido y necesito otro tipo de cable y un puerto paralelo avanzado, mientras el modo 0 trabaja con cables “parallel Nullmodem” estándard y con cualquier tipo de computadora.

La configuración de estos dispositivos y redes se discutirá a través de ejemplos de aplicación concreta en otros capítulos.

## 8.2. Protocolos

La transferencia de información en TCP/IP se realiza a través de paquetes IP, que pueden ser de clase TCP o UDP.

Ambos disponen de un encabezado (sobre) en el cual el destino y el remitente son indicados en forma de números IP, que son representados como cuatro números decimales con un rango de 0 a 255 separados por un punto. Ej: 192.168.3.45

Para usos especiales se realizaron modificaciones del estándard original, ejemplos son:

**ppp** Point to Point protocol. Un protocolo que se especializa en la transmisión de paquetes TCP/IP a través de canales intermitentes de un solo transmisor a otro, como pasa con líneas telefónicas. En este caso no hay requerimiento de enrutamiento (solo hay una posible ruta) por lo que se aplica compresión del volumen transmitido aprovechando la redundancia de los encabezados IP. ppp prove autentificación y negociación de múltiples parámetros de la transmisión, es un protocolo simétrico peer to peer, o sea, cada estación es independiente y autónoma y en ambos direcciones puede iniciarse el enlace.

ppp puede usarse favorablemente también para conexiones experimentales entre dos computadoras, p.ej líneas de transmisión “caseras”, equipos de infrarojo, ultrasonido, radio, etc.

**slip** Serial Line Internet Protocol. Es el predecesor de ppp e inferior en calidad. Sin embargo se utiliza a veces para conexiones virtuales, p.ej. existe un programa en Linux que marca una línea telefónica por demanda - diald, que utiliza slip como enlace de control.

**plip** es otro protocolo de punto a punto, que utiliza el puerto paralelo de una computadora para enviar y recibir datos. En este caso los paquetes no se transmiten en forma

> serial sino de cuatro en cuatro bits (nibbles) o en de ocho en ocho (bytes) según la capacidad del hardware. plip es la norma de desensamble/ensamble de los paquetes y las reglas de sincrónización - handshake entre las dos computadoras involucradas.

Los números IP en Internet son asignados por instituciones coordinadas (p.ej. `www.nic.ni`), para que ningunas dos computadoras tengan configurados el mismo número. Si configuramos un Intranet nosotros tenemos que cumplir esta tarea. Existen playas de números "privadas" que son reservados en Internet para la configuración de Intranets. Un rango que usamos en este libro para ilustrar nuestros ejemplos es: 192.168.0.0 - 192.168.255.255

Por ende vale mencionar, que los números IP no designan computadoras, sino dispositivos de red. Comunmente una computadora, especialmente si es de mesa, solo dispone de una conexión a la red, y por lo tanto es accesible solamente con un número IP. En este caso solemos identificar la computadora con su número IP.

Sin embargo imaginémonos la situación de una computadora de mesa con un modem conectado al Internet, a la cual conectamos una portatil via el puerto paralelo. En este caso la computadora de mesa dispone de dos números IP, una hacia el Internet que es asignada en el momento de conectarse via ppp y otra, que nosotros configuramos a la hora de instalar el interfaz plip hacia la computadora portatil.

## 8.3. Tabla de enrutamiento

En computadoras de mesa y muchas veces en servidores sencillos es suficiente despachar los paquetes de información de una manera fija - estática. Para esto, el núcleo de Linux dispone de una tabla de enrutamiento - routing table. La tabla de enrutamiento es una lista de especificaciones de rangos (playas) de direcciones IP, asociados con el dispositivo de red que se utilizará para despachar paquetes con destino en este rango. Eventualmente se puede configurar un "gateway" que es un número IP que designa una computadora capáz de remitir un paquete hacia otros segmentos de la red.

> rango-ip → dispositivo
>
> rango-ip → gateway →dispositivo

El comando "route" es utilizado para modificar y analizar la tabla de enrutamiento. Evocando este comando sin parámetro arroja la lista de rutas (routes) en la pantalla. "route add ..." agrega rutas y "route del ..." borra rutas de la tabla. La palabra "default" es una abreviación para una ruta por defecto que entra en función cuando no hay ninguna ruta establecida para una dirección IP específica.

En este momento vale mencionar, que el enrutamiento por número IP es un "enrutamiento externo", que permite establecer comunicación entre dos computadoras en una red ámplia. Hay también un "enrutamiento interno", que permite realizar una variedad de conexiónes lógicas entre estas dos computadoras. Esto se realiza a través de los "puertos" TCP. Prácticamente cada servicio Internet tiene asignado oficialmente un puerto, que no es nada más que un número de 16 Bit, que identifica un canal de comunicación, dentro

de un conexión física establecida entre dos computadoras. Entre todos los paquetes intercambiados se filtran para cada aplicación los paquetes correspondientes a su canal a través del número del puerto. Para poder utilizar el mismo servicio más que una vez, al comienzo de una conexión entre las dos computadoras se negocia en el puerto estándar un canal (puerto) libre, el cual se utiliza a continuación.

## 8.4. Servicios Internet

En el archivo /etc/services se relacionan nombre de servicios con su respectivo número de puerto. Este archivo es coordinado por el comité regulador de Internet y viene con asignaciones estándardizados, sin embargo puede ser ampliado o modificado al gusto del administrador del sistema. Casi todos los programas clientes y servidores tienen la opción de especificar puertos alternativos a los indicados en /etc/services, sin embargo esta opción se usará solamente en casos especiales.

Todos los servicios implementan una relación cliente servidor, es decir, al conectarse a una computadora (remota) mediante un programa 'cliente' se establece una comunicación con un programa 'servidor'. Al establecer una conexión el servidor en muchos casos le exige autentificación al cliente y le puede denegar el servicio según el resultado.El servidor en muchos casos implementa un "lenguaje" de comandos en texto claro (ASCII). Estos lenguajes son definidos en los documentos RFC – Request-For-Comment lo que en realidad significa propuesta para comentarios, sin embargo suelen a llegar a ser "normas".

He aquí algunos (pocos) de los servicios de red que se usan más frecuentemente.

**telnet** 'sniffer' en texto claro. Inclusive la clave del login se transmite sin encripción.

Telnet se puede utilizar para comprobar la disponibilidad de un servicio IP en un host mediante la especificación del puerto correspondiente.

**ftp** es un protocolo para la transferencia de archivos. Como tál permite el acceso al directorio home del usuario autentificado y puede transmitir archivos del y hacia la computadora remota (get archivo, put archivo). con mget y mput se pueden seleccionar archivos múltiples, con dir o ls se puede leer el contenido de un directorio. Ftp transmite en modo binario (image) o en modo ascii, hay que cerciorse siempre que se utiliza el modo binario, especialmente el cliente MS no lo tiene configurado por omisión.

Eventualmente (y en Internet muy común) se permite acceso a un usuario 'anonymous' al cual se le pide como clave su dirección de correo electrónico. A este usuario se le brinde acceso 'world' a un directorio especifico, donde se depositan archivos disponibles para la publicidad. ftp-anonymous crea una debilidad de seguridad en un servidor.

**http** es un protocolo para la transferencia asimétrica de archivos. La transferencia es solamente desde el servidor hacia el cliente. Este servicio se utiliza para la carga de páginas Web. Las solicitudes de los archivos se efectuan a través del envió de un 'URL'-unified resource locator. Los archivos se llaman recursos, ya que pueden

llevar información de cualquier tipo. El protocolo es mejor optimizado que ftp y no necesariamente se interrumpe la comunicación al ocurrir un error de transmisión.

**smtp** es el protocolo para el envío de correo electrónico. Smtp es un protocolo en texto claro y dado su carácter de transmisión de información confidencial el protocolo más problemático en cuanto a la privacidad. Esta situación ha llevado al desarrollo de técnicas muy maduras de encripción PGP y sus succesores GPG y otros. Smtp no solamente se utiliza para intercambio de mensajes entre humanos sino también para el acoplamiento de procesos técnicos y control de flujo. Smtp no provee un mecanismo de autentificación, ya que el enrutamiento del mensaje no se efectúa directamente sino a través de una meta-red de Servidores MX (Mailexchange). Un mensaje es injectado a la red en un servidor normalmente mediante un programa MUA (Mail-User-Agent) hacia un programa MTA (Mail Transfer Agent), el cual se preocupa por enrutarlo hacia su destino. En algún momento el mensaje llega a un servidor (MTA) que lo reconoce como mensaje local y lo deposita en su correspondiente apartado. Desde allí puede retirarse, otra vez con un MUA, o eventualmente con el protocolo pop (post office protocol) o imap. Entonces se reconocen tres fases de transferencia: MUA->MTA, MTA->MTA, MTA->Depósito (delivery).

**pop3** el Post-Office-Protocol versión 3, se utiliza para recoger remotamente, de un casillero electrónico los mensajes. La mayoría de los programas lectores de correo (MUA) utilizan este protocolo para retirar correo del servidor. También es un protocolo que prove poca proteción de privacidad de los mensajes.

**rpc** Remote-Process-control es un servicio para ejecución remota de procesos. Fue diseñado junto con una variedad de otros protocolos por Sun Microsystems, como el nfs e yp. Estos protocolos utilizan un sistema de redirección de puertos, lo que implica un tratamiento diferente a los otros protocolos mencionados. La seguridad e integridad de estos protocolos ya no concuerda con los estándardes de la investigación actual y se han diseñado varias alternativas, sin embargo son tán populares – ya que proveen servicios que son altamente solicitado en trabajo en grupo y computación distribuida – que hasta hoy no han sido reemplazados.

**nfs** Network-File-System es el protocolo que permite compartir directorios de una computadora con otra a través de una conexión TCP/IP. Ambas computadoras tienen acceso a los archivos compartidos y para ambos se presentan como archivos común y corrientes de Unix.

**otros** tftp, dhcp, bootp, ssh, xdmcp, rsync, cvs, dns, imap, nfs, yp, etc.

## 8.5. Configuraciónes ejemplares

### 8.5.1. Conexión telefónica a redes

Este tipo de conexión es altamente vulnerable, ya que se basa en una técnica de transmisión y conexión de baja calidad y robustéz en comparación con otras tecnologías. Además

hay varios niveles en los cuales pueden ocurrir problemas. Por eso vamos a analizar este ejemplo muy a detalle, aunque en la actualidad existen herramientas de configuración muy versátiles que simplifican sustancialmente la tarea de configuración y conexión.

A nivel de dispositivo podemos contar con un modem externo o interno.

### Modem externo

Este se conecta a través de un puerto en serie, o COM-Port. Hay que configurar la velocidad de conexión entre modem y computadora, así como los demás parámetros, mediante el comando setserial. Mientras en viejas distribuciones se utilizaron puertos /dev/cua* actualmente se considera el uso de /dev/ttyS* donde ttyS0 = COM1, ttyS1=COM2, etc.

Modem modernos pueden utilizar hasta la velocidad máxima (IBM-Compatibles 115200 Bd), pero hay modem que requieren una velocidad específica máxima, un valor seguro puede ser 9600 Bd, muy seguro: 2400 Bd. Esta velocidad solo tiene que ver con la velocidad de conexión telefónica en el sentido, que no se puede transmitir más rápido en la línea telefónica que en la conexión entre computadora y modem. Sin embargo puede haber una conexión mas lenta en el teléfono que en el cable serial, así que es recomendable usar velocidades reducidas solamente para pruebas.

Hay que asegurarse, que se utiliza un cable correcto. Las computadoras PC utilizan tradicionalmente un conector DB25 para el puerto paralelo y para el puerto serial. Estos se distinguen, en que el puerto serial usa pines (macho). Con el modelo PS2 se introdujo el uso de conectores de DB9 (9 pines) para el puerto en serie, que por cierto es contra la norma RS232C, pero viene siendo práctico para reducir el tamaño de las cajas de computadoras. En todo caso el cable debe tener en un lado un conector DB25 con pines que se conecta en el modem en un conector "hembra", y entonces queda por lo general solamente la elección entre uno o dos conectores seriales en la computadora. Eventualmente se requiere de un convertido DB9 a DB25 - este también existe en dos formas y hay que fijarse bien cual tipo es necesario. También recuerda conectar el cable al modem primero.

Hay cables "incompletos" que no tienen conectados todas las líneas. Se puede realizar una conexión serial entre dos modem (o computadoras) con tres hilos, sin embargo esto no es recomendado porque requiere un mecanismo llamado Software-Handshake (Xon-Xoff) que no es tan confiable y además reduce la velocidad de transmisión de datos. En efecto en el conector DB25 no se ocupan todos los pines, en el conector DB9 sí. El uso de Software-Handshake requiere una configuración especial y no lo contemplamos aquí. Si utilizamos un cable incompleto (o con algún hilo quebrado!) pueden haber problemas de conexión, que no son detectables instantaneamente.

En caso que no hay posibilidad de distinguir cual conector corresponde a cual puerto serial hay varios posibilidades. Lo más recomendable es abrir la caja y verificarlo físicamente. A veces en la tarjeta madre o en la tarjeta serial se puede leer el número del puerto asociado con el conector respectivo.

Un método seguro es la utilización de un probador de puertos seriales en conjunto con un software para controlar el puerto, p.ej. el emulador de terminales "minicom", kermit, sayon, u otro.

Habiendo detectado el conector no necesariamente identifica el dispositivo ttyS* ya que este puede cambiarse eventualmente mediante jumper o en el CMOS-Setup (BIOS) de la computadora, así que posblemente hay que referirse al manual de la tarjeta y/o al CMOS-Setup de la computadora. Los valores estándard para los puertos utilizados son:

| COM* | ttyS* | IO-Port | Irq |
|---|---|---|---|
| Com1 | ttyS0 | 0x3f8 | 4 |
| Com2 | ttyS1 | 0x2f8 | 3 |
| Com3 | ttyS2 | 0x378 | ? |
| Com4 | ttyS3 | 0x278 | ? |

En las computadoras modernas puede que el puerto USB o el puerto infrarojo utiliza un dispositivo serial (el secundario), así que en el CMOS-Setup se puede configurar a cual puerto correspondiente al dispositivo restante (el primario), y con eso determinar el ttyS* a usar para el modem.

### Modem interno

En este tipo de modems se ahorran los problemas con los cables, sin embargo es requerida la configuración del puerto y de la interrupción. Esto se realiza en modems “viejos” con conector ISA o EISA mediante el sistema PnP (Plug ’n Praise). La utilidad “pnpdump” arroja en la pantalla todas las posibles configuraciones para un modem PnP, en una forma que permite de manera fácil crear un archivo de configuración (p.ej. /etc/isapnp.conf). Este debe cargarse antes de usar setserial mediante la utilidad “isapnp”, (p.ej: isapnp /etc/isapnp.conf)

A veces se requiere reinicializar la computadora para este paso, inclusive puede ser necesario apagarla. No culpen al Linux por eso, hablen con ...

Modems internos con conector PCI también utilizan PnP pero “de lo bueno” y su configuración se puede detectar con la utilidad “lspci”, o con el comando “cat /proc/pci” (en núcleos con versiones inferiores de 2.4.0).

Generalmente se utiliza la velocidad de 115200 Bd para modems internos, ya que es un valor no relevante, pués no existe un conexión serial entre la tarjeta-modem y la computadora; sin embargo se tiene que usar un valor compatible con el driver serial.

### Conectividad con el modem

Una utilidad bastante sencilla para comprobar la funcionalidad básica de un modem es el programa minicom, que permite conectarse al modem en modo “terminal”. Cada tecla que se aprieta se envia mediante el puerto serial al modem, cada caracter que el modem remite al puerto serial se visualiza como caracter en la pantalla.

Como root tiene que configurarse el puerto a usar (C-A O), es importante salir (C-A X) y volver a entrar de minicom para que los cambios tengan efecto (al menos conmigo). Al volver a entrar normalmente ya se puede apreciar un diálogo entre minicom y el modem.

La mayoría de los modem utilizan comandos "Hayes" para su control. Estos inician con los caracteres "AT" (ATtention). El modem refleja cada caracter y remite una respuesta, si el comando es exitoso la respuesta es "CR-LF", o sea, el cursor de la terminal pasa a la siguiente línea.

A continuación una lista de comandos básicos:

| AT* | Parámetros | Significado | Comentarios |
|---|---|---|---|
| ATZ | | Reinicializar Modem | |
| ATH | 1/0 | "levantar"/colgar el telefono | A veces solo se puede usarATH y ATH0 |
| ATD | # telefónico | Marcar un número (dial) | El modem espera estableceruna conexión de datos. Estose indica con "CONNECT".En caso que no se puede conectardespués de un tiempo se recibeun mensaje de error. |
| ATA | | Contestar (Answer) | Si entra una llamada el modemenvia el texto "RING" altermnial. Si se contesta el modemespera sincronizarse con unmodem remoto y señaliza estehecho con "CONNECT". |

El modem trabaja en uno de dos modos: en modo de "comando" podemos enviar los comandos al modem, pero después de establecer una conexión el modem trabaja en modo "transparente", donde todos los caracteres se envian al modem remoto es decir a la computadora remota respectivamente ya que el modem remoto también está en modo transparente. De este modo se puede regresar al modo comando por falla de conexión: interferenica, interrupción de la línea etc., o con una secuencia específica de caracteres.

### Establecer un enlace

El primer paso para conectarse a un servidor internet es marcar el número, lo que efectuamos con el comando "ATD" seguido por el número del servidor. El modem marca primero el número y si la línea no está occupada se puede esperar que al otro lado contesta el modem del servidor. En este momento los dos modem intentan sincronizar la velocidad y el protocolo de transmisión sobre la línea telefónica, un proceso que no siempre es exitoso, especialmente con altas velocidades de transmisión y cuando la línea telefónica sufre interferencias.

Una vez que se "entienden" los modems lo notifican a la computadora; normalmente con el texto "CONNECT", y a veces seguido por la velocidad y los parámetros de la conexión, y pasan inmediatamente al modo transparente.

Ahora le toca el turno al proveedor del servicio. En la actualidad prácticamente todos los proveedores de servicios Internet via la línea telefónica utilizan el protocolo PPP para establecer el enlace de red, y nosotros vamos a enfocar en este.

Aunque PPP incluye un método de autentificación, muchos proveedores requieren una autentificación en texto pleno, normalmente con un diálogo como lo escribimos en los capítulos iniciales (4.1). Podemos realizar la autentificación manualmente, pero también hay utilidades para automatizar este proceso, dos de ellos (los más usados) son "chat" y "runscript". Este último es parte de minicom. Para curios@s refiero a las páginas manuales de estos programas y a las noches interminables que pueden pasar cuando empiezan a experimentar con ellos. Sin embargo hay que mencionar una cosa: La autentificación en texto pleno es un atentado contra la seguridad y es preferible cambiar de proveedor antes de acceptar esta clase de anacronismo. La utilización de scripts de conexión requiere por lo general escribir la clave de una (o varias) cuentas en un archivo texto y por lo tanto directamente legible para quien tenga acceso al archivo. Por lo tanto aumentamos también con esto la vulnerabilidad de nuestro sistema.

En algunos casos el proveedor configuró el servidor de tal manera, que después de la autentificación se lanza inmediatamente el servidor ppp, en otros casos nosotros tenemos que iniciarlo desde una línea de comando remota, por ejemplo con el comando "ppp<Enter>". En ambos casos hay que lanzar a continuación inmediatamente el enlaze ppp localmente. Si nos estamos conectando con minicom u otro emulador de terminal tenemos que terminar este sin reinizialización del modem o corte del enlace. Terminar: porque el puerto serial va siendo bloqueando por cada aplicación que lo occupa, para que no haya conflictos en su uso; Sin reinizializar: normalmente los programas que usan un puerto serial modifican sus parametros al usarlo y tratan de reestablecer los parametros anteriores antes de ceder el dispositivo. En el caso de minicom se puede lograr estas dos condiciones con la secuencia de teclas: "C-a q".

### Dispositivo virtual de red: ppp

La conexión transparente entre las dos computadoras ahora se utiliza para realizar una conexión TCP/IP, mediante dos dispositivos de red, que se crean transitoriamente mediante el daemon pppd (en caso de una máquina Unix). Este es un programa que lee caracteres en su stdin y escribe en el stdout. Si lo lanzamos, especificando como uno de los parámetros de la línea de comandos un dispositivo de caracteres, pppd utiliza este dispositivo como stdin/stdout, en nuestro caso le especificamos entonces el puerto serial donde está conectado el modem.

pppd realiza la conexión con el pppd remoto en tres etapas:

1. Establecimiento del enlace via LCP (Link Control Protocol) y configuración de diferentes parametros de la red como MTU, número IP de ambos participantes, servidor de nombres, modos de compresión de datos/encabezados, etc.

2. Autentificación mutua. ppp es un protocolo simétrico y en principio ambas computadoras tienen que comprobar el derecho de conectarse a la respectiva otra parte. En el caso de un enlace hacia un proveedor de servicios internet normalmente el

"cliente" no puede exigir autentificación al "servidor", un hecho que tiene que reflejarse en la línea de comando del pppd o en los archivos de configuración.

3. Intercambio de paquetes TCP/IP.

pppd puede ser configurado de una manera que le permite asignar al lado opuesto uno de un rango de números IP (eventualmente el rango solo consiste de un número), esto se hace cuando pppd es usado como "servidor". En "nuestro" lado local podemos acceptar la configuración propuesta remotamente, pero también se podría insistir en un número IP fijo. Otro parámetro importánte para el daemon local es, "defaultroute" que inserta en la tabla de enrutamiento de paquetes IP una salida por defecto a través del dispositivo transitorio que ha creado el daemon (ppp0, ppp1, etc.). Al cerrar la conexión la ruta por defecto original se restablece. De nuevo: si utilizamos ppp para conexiones, por ejemplo via un enlace radio hacia un subdominio de nuestra red comercial intranet, no se establecería una ruta por defecto, sino solamente para este subdominio, por lo cual la configuración sería diferente.

Existen dos formas de autentificación: PAP y CHAP.

**Password Autentication Protocol** el daemon local envia un nombre y una clave hacia el daemon remoto. Este por su parte compara los dos datos con todas las líneas del archivo /etc/ppp/pap-secrets (las claves en ppp son llamados secretos). Se se encuentra una línea correspondiente se autoriza la conexión, de otra manera el enlace se cierra.

La clave puede archivarse en /etc/ppp/pap-secrets en forma encriptada (unix-crypt) y se puede exigir que solamente secretos encriptados sean acceptados (en la configuración del daemon remoto - pruebelo con su proveedor). Crypt hoy en día no es una encripción fiable, cualquier computadora común puede descubrir un secreto encriptado con unix-crypt dentro de minutos.

**Challenge Handshake Autentication Protocol** Con este método nunca se intercambia el secreto en línea. El daemon remoto inicializa la autentificación, enviando su "nombre" y un número al azar - el "reto" (challenge). El daemon local produce un código único[1] desde el reto y desde la clave que encuentra en el archivo /etc/ppp/chap-secrets en la línea que contiene el nombre del daemon remoto. Este código único es remitida al daemon remoto, quien es el único que puede comprobar la validez, porque solamente él comparte el "secreto" y puede crear el mismo código único para la validación de la identidad del daemon local. CHAP usualmente repite el proceso de autenticación constantemente en ciertos intervalos. De esta manera se intenta evitar un ataque man-in-the-middle.

A continuación una línea de comando típica, que puede usarse experimentalmente para conectarse con un daemon ppp remoto:

[1] Pués, casi único. En realidad se produce un hash, o como dicen los matemáticos españoles: una función de dispersión.

pppd /dev/ttyS0 modem crtscts defaultroute noauth

Para averiguar problemas en una conexión se pueden utilizar las siguientes opciones:

**debug** escribe información sobre todos los paquetes de control en el syslog. Esta información puede rastrearse entonces con: "tail -f /var/log/syslog" (Debian), o "tail -f /var/log/messages" (RedHat).

**logfile nombre** escribe la información de purificación también al archivo "nombre".

**show-password** muestra la clave en texto pleno en el archivo log. Redunda decir, que esta opción preferiblemente no se utiliza, mucho menos de manera constante.

**kdebug #** muestra en el archivo log también información de paquetes IP desde el núcleo.

Los archivos que contienen los secretos para PAP y para CHAP tienen la misma estructura:

```
# Comentario
# Nombre_de_cuenta Nombre_de_Computadora Secreto
12310ka             ikarus          el_sol_quema
lodemas             *               3i4fe0es65
# un asterisco es válido para cualquier nombre de computadora
# en todas las conexiones menos con (ikarus) se usara el nombre lodemas
# con su secreto (encriptado)
```

Por fín hay que saber, que por lo general pppd es ejecutado sin opciones de la línea de comando, ya que todas las opciones se pueden grabar en los archivos de configuración, en primero lugar en /etc/ppp/options.

### El mundo real es más dulce

En muchos casos desde una computadora no solo se conecta a un proveedor, sino eventualmente a varios, o se utiliza por lo menos diferentes números telefónicos para el enlace. Sea así, o haya solamente una conexión, no es factible que todo el mundo aprenda utilizar minicom, syslog y pppd desde la línea de comando para poder configurar su conexión telefónica a redes. A continuación una lista de programas que se han hecho para facilitar la configuración y la conexión, pero primero algunos aspectos desde el punto de vista de su utilización.

Se requiere de la *configuración* que se pueda introducir las características para una conexión. Estas se dividen en dos grupos: el dispositivo a usar: puerto serial, velocidad, modem, y los parámetros de conexión: número telefónico del servidor, nombre de la cuenta, secreto, y parámetros: autentificación previa, CHAP/PAP entre otros. Tienen que poderse manejar diferentes configuraciones, que se identifican y manejan a través del "nombre de la conexión".

Después se necesita utilidades que facilitan el *control* de la conexión: establecerla, cortarla y poder saber si se está conectado o no, asi como el tiempo transcurrido, la cantidad de información enviada y recibida, y eventualmente el costo incurrido:

**modemtool** configuración: de RedHat, con un interfáz gráfico/X que es muy intuitivo en el uso.

**wvdial nombre** configuración: (elaborado por World Vision - wv) que detecta y configura el/los modem/s utilizado/s automáticamente y crea archivos de configuración llamado “nombre”, que puede completarse manualemente con un editor de texto para agregar los datos de autentificación.

**pppconfig** configuración: la utilidad que prove Debian, cuenta con un interfáz gráfico/texto. Fácil de usar. Hay una conexón por defecto llamado “provider”

**pon, poff** control: son scripts que establecen/cortan la conexión con “provider”. Se puede especificar otro nombre de conexión en la línea de comando.

**masqdialer** es un daemon en forma cliente/servidor, que permite el control de un modem para la conexión telefónica desde una red local. El servidor: mserver tiene que configurarse en una computadora donde ya existen todas las conexiones con su nombre. Estas (y otros datos mas) se le comunica a mserver a través de archivos de configuración. Un “dial”-cliente manda comandos al mserver via TCP/IP, y el mserver le prove información estatística sobre la conexión (tiempo, volumen, velocidad). Hay clientes texto, gráficos/X, gráficos/gtk, etc. masqdialer permite establecer permisos de acceso por computadora y por usuario por lo que puede configurarse bien quien gasta teléfono en qué servidor y quien no.

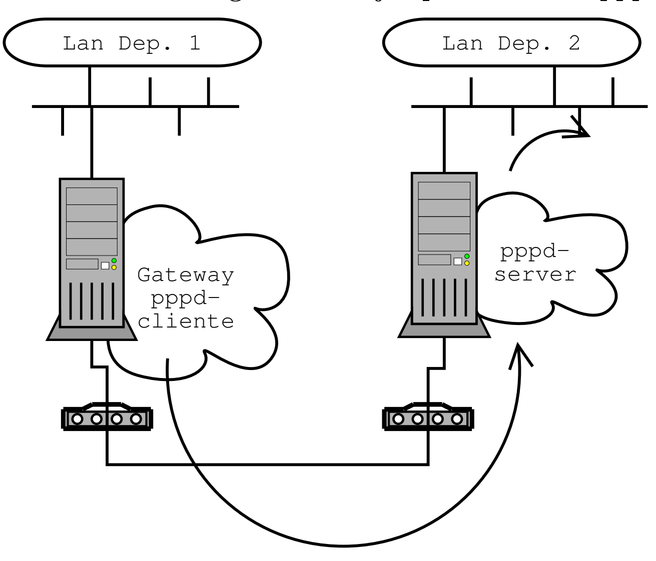
**Apletts** Hay varios utilidades para X-Windows, que consisten en esencial en un botón y una “luz”. Haciendo Click en el botón establece/corta el enlace. La luz cambia de color si se está conectado.

### Desde adentro

algunos conceptos que hacen pppd interesante para redes locales.

**ip-up/ip-down** Muchas veces se requiere realizar algunas tareas específicas al momento de establecer un enlace con la red: en primer lugar para establecer rutas especiales, reglas firewall para el interfáz u otras configuraciones especiales de la red, pero también por ejemplo para enviar automáticamente todo el correo en cola, y para bajar correo o noticias del internet, cuando el enlace se establece. Para esto pppd puede ejecutar un script al establecer el enlace, y otro al cerrarlo. En las instalaciones modernas este script accede a un directorio `/etc/ppp/ip-up.d` y `/etc/ppp/ip-down.d` en el cual pueden ubicarse de manera ordenada diferentes scripts, que se ejecutan en orden específico (alfabético). El mecanismo es parecido al que se utiliza con “init” del sistema V de unix. De esta manera cada paquete de software puede agregar y quitar tareas según necesidad y sin interferir con otros paquetes.

Figura 8.1.: Ejemplo de uso de ppp para una red local




**/etc/ppp/peers /etc/chatscripts** Para establecer un enlace tienen que compaginar el discado y la configuración del pppd. En el sistema Debian se encuentran los scripts chat para la primer tarea bajo el directorio /etc/chatscripts, y la configuración del pppd bajo /etc/ppp/peers, bajo el mismo nombre. pppconfig modifica ambos archivos al configurar un enlace nombrado. Eventuales cambios manuales pueden realizarse allí mismo.

**pppoe** es el acrónimo para PPP Over Ethernet. En este caso se utiliza una conexión Ethernet para el enlace transparente.

**pptp** es el acrónimo para Point to Point Tunnelling Protocol. Es usado para establecer una conexión bajo DSL/ADSL ([Asyncronios] Digital Subscriber Line).

**ippp** es el protocolo ppp utilizado por conexiónes ISDN (Integrated Services Digital Network).

**y otros** ppp es muy versátil para establecer conexiones TCP/IP de punto a punto. Por lo tanto es utilizado muchas veces para redes WAN. El "costo" de desarrollar la tecnología entonces se reduce a la tarea de establecer un canal transparente. Nosotros podemos hacer lo mismo. Por ejemplo pueden enlacarse dos departamentos distantes de una institución mediante la planta telefónica, o eventualmente con un simple cable de dos hilos que simula una línea telefónica (requiere alimentación electrica). En el último caso nos podemos valer del hecho, que los modem intentan establecer una conexión con el comando "ATD", aunque no especificamos un número telefónico. Por otro lado, con el comando "ATA" podemos iniciar la negociación de velocidad, aunque no se reciba la señal de marcado ("RING"). Para que el modem primero no corte la línea por falta de tono ("NO DIALTONE") se

puede omitir esta condición con el comando "ATX3". En la imagen 8.1 se muestra solamente una rama de la configuración. La computadora a la izquierda utiliza (al arrancar) el modem y ppp para tratar de conectarse a la computadora a la derecha. Todas las computadoras en la Lan del departamento 1 utilizan esta computadora como Gateway para la Lan del departamento 2. Si el enlace no se puede establecer, la computadora a la izquierda pone el modem en modo answer ("ATA") con un timeout infinito. La misma configuración se realiza con la segunda computadora.

**demand dialing, dialdcost** El programa diald (dial-daemon), utiliza un enlaces virtual (en este caso SLIP), para detectar si alguna computadora de una LAN solicita una conexión con Internet. En este caso establece un enlace ppp con el modem y enruta la conexión a través de ella. Si durante un cierto tiempo no hay tráfico la línea telefónica se corta. De esta manera se puede tener un enlace casi contínuo, sin tener que pagar el costo de una línea dedicada, los usuarios solamente tienen que tener un poquito de paciencia para permitir que se establece el enlace.

Junto con configuraciones de este estilo vienen práctico utilidades que permiten sumar el costo total de la conexión, llevar el control de cuando se establece y cuando se corta la conexión, y hasta que usuari@ ha consumido cuánto tiempo de conexión. El programa dialdcost y ip-account se occupan de estas tareas.

### 8.5.2. Syncronización de una Portatil con una PC de Escritorio

En vez de utilizar disquetes, o unidades externos de almacenamiento de datos como ZIP-Drives etc. para copiar datos entre una computadora portatil y su PC "de casa" y de oficina se pueden enlazar mediante una conexión de cable. Hay soluciones en MS-Dos y Windows, que siguen un esquema de maestro – esclavo donde una parte inicia la comunicación y domina el proceso de copiado. El programa posiblemente más antíguo es "kermit", que utiliza un puerto serial en ambas computadoras. El puerto serial tiene la desventaja de una velocidad baja y la serialización de los datos: un caracter requiere aproximadamente 10 ciclos para su transmisión.

Una salida de este cuello de botella es el uso del puerto parallelo, desvirtuando las funciones de las líneas de control. La idea es, utilizar cuatro líneas de datos (de los ocho disponibles) como salida parallela, y cuatro de las líneas de señalización como entradas. Estas líneas normalmente usa la impresora conectada para señalizarle a la computadora fallas como la falta de papel o su estado para recibir datos (occupado/listo). Las "entradas" se conectan a traves de un cable especialmente preparado a las "salidas" de otra computadora. Ambas computadora por supuesto utilizan un programa especial, con el cual pueden intercambiar datos serializados en nibbles. Haciendose común este arreglo fue tomado en cuenta por los diseñadores del hardware de las computadoras y actualmente hay puertos parallelos que pueden reconfigurar las ocho líneas de datos como entrada y/o salida y se pueden utilizar las líneas de señalización para el control. El hardware permite de esta manera a veces inclusive transferencia de bloques grandes de datos mediante DMA, aliviando la CPU del trabajo de controlar la conexión. Tambien esta forma de conexión utiliza un cable especial, de ningúna manera puede usarse un cable de impresora

para interconexión de dos puertos parallelos.

Linux provee una alternativa al enfoque maestro – esclavo, proveyendo un dispositivo de red a través del puerto parallelo. El protocolo utilizado permite el transporte bidirecional de paquetes TCP/IP a través del puerto parallelo y se llama PLIP: Parallel Line Interface Protocol. El driver (módulo de núcleo) y el nombre del dispositivo de red llevan este mismo nombre.

Es recomendable que el driver se compilado como módulo de núcleo, y no integrado fijamente a este, ya que de esta manera se puede cargar y descargar de la memoria y liberar el puerto parallelo después de usar la conexión PLIP. Si se compila fijamente, no será posible utilizar el puerto parallelo para otra cosa, como por ejemplo la impresión[2].

### 8.5.3. Preparación del puerto paralelo para PLIP

PLIP requiere el uso de la interrupción hardware para el puerto paralelo. Normalmente este no está activado, ya que el uso del puerto paralelo para la impresión no lo requiere, y las interrupciones son un recurso muy valioso como para ocuparlo innecesariamente. Por lo tanto necesita activarse la interrupción, o a la hora de cargar el módulo del driver del puerto paralelo (parport_pc), o posteriormente mediante el sistema virtual de archivos 'proc'.

> Notese, que hay (por lo menos) dos cambios esenciales entre las versiones de núcleo antes de y apartir de la versión 2.4. respectivamente. El primero es la ubicación de los módulos de núcleo en una jerarquía de directorios en /lib/modulos, y la segunda es la estructura de la información del núcleo en el sistema virtual de archivos /proc. A continuación se presenta la información pre-2.4. Para núcleos 2.4 y adelante hay que cambiar los parámetros.

**driver:** insmod /lib/modules/*version-del-núcleo*/misc/parport_pc io=*puerto* irq=*número*

**proc:** echo *numero* /proc/parport/*número-puerto-paralelo*/irq

donde:

*número* es el número de la interrupción que utiliza el hardware y
*puert—s7s6s9o* la dirección de I/O (entrada/salida) correspondiente.
En la siguente tabla se dan algunos valores estándard:

| Puerto | I/O | Irq | nombre en dos |
|---|---|---|---|
| /dev/lp0 | 0x378 | 7 | LPT1: |
| /dev/lp1 | 0x278 | 5 | LPT2: |
| /dev/lp2 | 0x3bc | - | En tarjetas monocromaticas |

[2]En los núcleos Línux más modernos ya existen mecanismos para el uso múltiple del puerto paralelo.

A esta altura tenemos un dispositivo de red disponible (se puede visualizar en /proc/net/dev), normalmente llamado “plip0”. El próximo paso es:

### 8.5.4. Configuración del enlace TCP/IP

Por supuesto tiene que efectuarse el anterior paso con éxito en las dos computadoras que se pretenden comunicar. A continuación hay que configurar en ambas el interfáz plip0 para que pueda comunicarse con la otra computadora. En la mayoría de los casos el enlace será temporal por lo que utilizamos números IP privados. Si queremos conectar una portatíl en diferentes computadoras de mesa, p.ej: en casa, en el trabajo, en la universidad, donde un cliente, etc. una posible configuración es, configurar en todas las computadoras de mesa el mismo número IP, digamos que sea: 192.168.255.254. La portatíl se configuraría con un número fijo, sea: 192.168.255.253, y se puede facilitar el trabajo dandole un nombre a estos números IP en el archivo /etc/hosts, p.ej plippc para las computadoras de mesa y pliplap para la portatil respectivamente.

Los enlaces plip tienen que configurarse con la opción punto a punto:

```
ifconfig plip0 192.168.255.253 pointopoint 192.168.255.254
```

en la portatil y

```
ifconfig plip0 192.168.255.254 pointopoint 192.168.255.253
```

en la(s) computadora(s) de mesa respectivamente.

Para más completo aquí una sección del archivo /etc/hosts, que sería iqual para todas las computadoras

```
192.168.255.253 pliplap
192.168.255.254 plippc
```

Una configuración ejemplar que funcionaría con la mayoría de las computadoras actuales puede ser:

```
echo 7 >/proc/parport/0/irq
modprobe plip
ifconfig plip0 192.168.255.253 pointopoint 192.168.255.254
```

Este segmento de comandos se puede grabar en una archivo script, p.ej. llamado plipon y ejecutar cuando se quiere conectar las dos computadoras. Obviamente hay que modificar la última línea de acuerdo a qué máquina se trata.

### 8.5.5. Observaciones adicionales

En caso que no se puede iniciar el uso del puerto como interfáz plip, puede ser que hay trabajos de impresión pendiente y el daemon de la cola de impresión “lpd” está occupando el puerto. Este hecho se puede revisar por ejemplo con el comando lpq, o lpstat. Si hay trabajos en cualquier cola de impresión que utilize el puerto de impresión en cuestión (revisar /etc/printcap y las entradas :lp=<puerto>:) tienen que terminarse o cancelarse estos primero.

El puerto paralelo puede existir con varias capacidades:

**SPP** es el modo estándard compatible con *todas* las computadoras PC (Standard Parallel Port)

**EPP** es un modo mejorado (Enhanced Parallel Port), en el cual la computadora puede leer informaciones de identificación de una impresora conectada (y compatible con IEEE 1284)

**ECP** prove mas “capacidades” (Extended Capabilities Port)

Si las dos computadoras disponen de puertos paralelos con dispositivos mejorados se puede posiblemente utiliza el modo de transferencia de un byte completo llamado también Modo 1. Para esto se requiere de un cable especial, que solamente se debe utilizar con las dos computadoras configurados para este modo, ya que de otra manera podrían causarse daños en los dispositivos de hardware. En resumen: El modo estándard, o Modo 0 utiliza un cable “especial”, llamado cable “Parallel-Null-Modem” (que es el mismo que utiliza LapLink para MS-DOS/Windows), y que no puede dañar al hardware (del puerto de impresión), pero por supuesto no puede usarse para imprimir. Este modo es lento ya que solo transmite un nibble - medio byte a la vez. El modo más rápido, Modo 1, requiere de: puertos mejores que SPP en ambos lados, un cable “especial” y que las computadoras estén configuradas ambas en Modo 1. La información como hacer estos cables se encuentra p.ej. en los archivos fuente del núcleo en el subdirectorio Documentation/networking/PLIP.txt

### 8.5.6. Intranet con LAN/Ethernet

Si estamos planificando una red local podemos servirnos en muchos casos de la tecnología Ethernet, que permite una conexión rápida, de bajo costo y con un soporte tecnologíco bien fundamentado. Aunque hay diferentes formas de cableado se han establecido dos en especial: Thin-Ethernet, con un cable coaxial RG50 y conectores BNC (parecido pero diferente a los cables de antena de la televisión), y RJ45 o UTP, con cables de diez hilos y conectores parecidos a los conectores americanes de teléfono, pero mas anchos.

Este último cableado ha ganado mucha importancia a la medida que se han bajado los costos para un dispositivo llamado “Hub” o concentrador, que es necesario para interconectar las máquinas en esta configuración.

Cada computadora necesita una tarjeta Ethernet, también llamado adaptador de red Ethernet o NIC - “Network Interface Card”, que en el caso del cableado UTP está co-

Figura 8.2.: Red local Ethernet con cableado coaxial

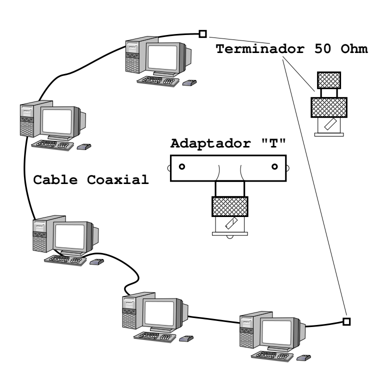
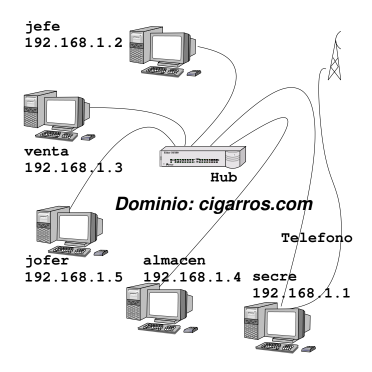



Figura 8.3.: Red local Ethernet con cableado UTP



nectada con el concentrador, en el caso del cable coaxial con dos máquinas “vecinas” a través de un adaptador “T”.

Vamos a asumir resuelto el cableado y la interconexión de todas las computadoras conectadas a la red, entonces necesitamos un gestor para la tarjeta de red, que facilita la comunicación entre el tráfico de información en la red y el núcleo del sistema operativo Linux (o cual sea).

Cada tarjeta Ethernet tiene un número único, que le está programado al momento de la producción. Cuando la trajeta transmite un paquetes de información binaria hacia la red, este es marcadoscon este número único, el NIC-number, o “hardware address” - la dirección hardware. Este número es importante en primer lugar para la sincronización y aseguración de la información entre las diferentes tarjetas que participan en la red. En escasos casos se requiere, pero en estos es recomendable pegar en cada computadora como referncua una etiqueta con el número NIC de la tarjeta Ethernet, el cual consiste de ocho números hexadecimales de dos cifras separado por un doble punto “:”.

### Gestor de la tarjeta Ethernet

Existen grandes variedades de productos de tarjetas Ethernet, que se diferencian básicamente por

- la forma de configuración
- el conector o bus con el cual se comunican con la computadora
- el tipo de circuito integrado que implementa la transformación de datos desde y hacia la red.

Tarjetas antiguas utilizan puentes para configurar los recursos: interrupción y dirección de Entrada/Salida. Hay tarjetas que sustituyen los puentes por una configuración programable en una memoria no-volátil EERAM. Estos requieren un programa especial para su configuración, que muchas veces solo está disponible para MS-DOS o eventualmente Windows. En estos casos es útil tener una vieja máquina con MS-DOS, una partición pequeña que puede arrancar con MS-DOS y contiene este programa de configuración, o una configuración especializada del emulador MS-DOS.

Algunas tarjetas son configurables mediante el interfáz ISA-PnP, para el cual hay soporte completo en Linux.

Hasta ahora solo estuvimos hablando de tarjetas para el bus ISA. Todas las tarjetas con bus PCI alivian el problema de la configuración porque está tecnología permite una autoconfiguración mediante el BIOS o mediante el núcleo del sistema.

Ente las tarjetas para el bus ISA son los más conocidos:

**NE2000** y compatibles, con el chip 8039

**3C50x** de la empresa 3Com, donde x puede ser un número de 1 a 9.

**Tulip** de la empresa Dec

Para el bus PCI que en la actualidad es más común, y para tarjetas de red integradas en la tarjeta madre (que utilizan la norma PCI para su comunicación con la CPU):

**NE2000-PCI**

**3C59x** de la empresa 3Com.

**SiS900** de Silicon Integrated Systems

**Davicom** con el chip DM9102 y parecidos

**RTL8139** de Realtek.

El gestor de la tarjeta Ethernet tiene que ser soportado por el núcleo del sistema operativo, o como parte estática o como módulo. Normalmente los núcleos de distribución contienen todos los gestores en forma de módulo, a continuación una lista de los nombres de módulos en el orden de mención de las tarjetas arriba:

> ne, 3c501 (p.ej.), tulip, ne2k-pci, 3c59x, sis900, dmfe, 8139too

Estos tienen que cargarse, y en el caso de las tarjetas ISA con los parámetros respectivos de interrupción y dirección, para que el núcleo disponga de ellas. Supongamos una red (con computadoras viejas) con tarjetas compatibles con NE2000. Para facilitar nuestro trabajo garantizamos, que en todas las computadoras podemos configurar las tarjetas para la dirección 0x340 y la interrupción 10.

Para cargar el gestor necesitamos ejecutar el comando:

```
modprobe ne io=0x340 irq=10
```

Esta configuración tiene que establecerse siempre al inicializarse el sistema operativo. Diferentes distribuciones de Linux tienen diferentes mecanismos para eso. En el sistema Debian se agregaría una línea

```
ne
```

sin opciones al archivo `/etc/modules` para cargar al iniciar este módulo. Los parametros se establecen a través del paquete modutils, creando un archivo `/etc/modutils/ne` (o /etc/modutils/ethernet), con la siguiente línea:

```
options ne io=0x340 irq=10
```

A continuación hay que ejecutar (una sola vez) el comando

```
update-modules
```

para incorporar las opciones en el archivo `/etc/modules.conf`.

En otros sistemas se utilizará eventualmente una herramienta gráfica para agregar el "dispositivo" con sus parámetros al sistema, o se edita directamente el archivo `/etc/modules.conf` con la línea indicada, y el archivo `/etc/rc.d/init.d/network` o `/etc/rc.d/init.d/modules` para ejecutar "`modprobe ne`".

### Direcciónes IP

Aunque en una red estrictamente Intranet no hay necesidad de usar números IP específicos es recomendable utilizar siempre números de los rangos privados, por ejemplo del rango: 192.168.1.0-255

Este es suficiente para una red Internet en base a Ethernet, pués esta tecnología no permite conectar más que 64 computadoras. Los números que terminan en 0 y en 255 no pueden usarse, porque son reservado para mensajes especiales de la red.

El número que termina en 1 se le asigna preferiblemente a la computadora que funge como enlace con otros redes (gateway), si es que hay un enlace; y esta computadora eventualmente también se utiliza como servidor primario de nombres (DNS).

Entonces supongamos que tenemos cinco computadoras en una oficina, de las cuales la de la secretaria se utiliza para conectarse por teléfono con Internet. Les asignamos los números IP 192.168.1.1 (secretaria), -2, -3, -4 y -5.

Asignamos un nombre de dominio a nuestra red, que puede consistir de un solo nombre, pero decidimos usar un nombre similar a las convenciones del internet: `cigarros.com`, ya que nuestra oficina ficticia es de la empresa "Cigarros del Caribe", también ficticia.

Cada una de las computadoras necesita un nombre de computadora *hostname*, que la identifica, y que probablemente ya ha sido establecido a la hora de instalar el sistema operativo. Hay distribuciones de Linux, que no molestan al/a la usuario/a en el momento de la instalación y le ponen el nombre "localhost", o algun nombre fijo. En el momento de crear una red, no es recomendable tener varias computadoras con el mismo nombre, aunque se diferencian por su número IP.

Por lo general no es recomendable darle nombres secuenciales a computadoras en red, como por ejemplo pc01, pc02, al menos que se trata de una sala de cómputos, o una serie de cajas electrónicas (caja1, caja2, etc.) donde la númeración de la computadora corresponde a un esquema numérico físico. Tampoco es de utilidad general ponerle a la computadora el nombre de la persona que la utiliza, porque esta puede cambiar probablemente antes de cambiar la computadora, y al menos que se aprecia la confusión, requiere entonces una reconfiguració de la máquina. Un sistema practicable es, buscar una categoría de objetos y utilizar nombres de elementos concretos, ejemplos:

**Ciudades** roma, napoles, venecia, pisa, gubbio

**Felinos** tigre, pantera, leon, gepardo, puma

**Maderas** roble, caoba, pino, granadillo, ceibo

**Pintores** davinci, vangogh, miro, durer, picasso

las posibilidades son infinitas, la idea es, darles un contexto a las y los usuarias/os, para memorizar más facilmente cuál computadora se llama cómo. Si las computadoras en una red son de diferentes categorias en cuanto a, modelos, uso o plataforma conviene usar diferentes categorias para sus nombres también. Por ejemplo pueden llamarse todas las computadoras con nombres de plantas; las que son para trabajos de secretaría llevan nombres de uvas, las que son para la programación de software llevan nombre de arboles,

las que se usan para tareas auxiliares (gateways, enrutadores, servidores de archivos o bases de datos) tienen nombres de hierbas.

En nuestra red fictica usamos como nombres las funciones de los empleados que los usuan:

```
jefe, secre, venta, almacen, jofer
```

En nuestra red pequeña solo tenemos un interfáz de red por computadora, por lo cual asociamos directamente su númerio IP con el nombre de la computadora. En redes complejas donde una computadora puede tener varios adaptadores de red el nombre *hostname* de la computadora no necesariamente está asociado con uno de ellos. Cada tarjeta de red tiene un número IP, por lo tanto una computadora que tiene varias tarjetas de red es accesible "conocida" por cualquiera de ellos.

En nuestra red pequeña inicialmente creamos un archivo /etc/hosts, para dar a conocer la asociación entre nombres de computadoras y números IP, lo cual es por ende la identificación de la computadora en red:

```
127.0.0.1 localhost
192.168.1.1 secre.cigarros.com secre
192.168.1.2 jefe.cigarros.com jefe
192.168.1.3 venta.cigarros.com venta
192.168.1.4 almacen.cigarros.com almacen
192.168.1.5 jofer.cigarros.com jofer
```

Este archivo tiene que copiarse a todas las cinco computadoras, con lo cual cada computadora está "conocida" con su nombre hostname y con su nombre completo (fqdn: fully qualified domain name).

Notese, que efectivamente cada computadora tiene dos (2) interfazes de red: el "loopback", llamado localhost y el interfaz ethernet. El interfaz loopback se utiliza como simúlacro para mandar información entre diferentes aplicaciones de red en la misma computadora. Ya que mensajes dirigido a la misma computadora no tienen que salire a la "publicidad" de la red no es necesario, que cada "loopback" en la red tenga un nombre diferente y como convención se utiliza el número IP 127.0.0.1 para el en todas las computadoras.

El número IP es asociado con su interfáz de red correspondiente a través de la utilidad de sistema `ifconfig`. El gestor para la tarjeta Ethernet al cargarse en la memoria revisa los dispositivos presentes (o indicados mediante los parámetros), y les asigna en el orden de detección símbolos de dispositivo. En el caso de las tarjetas Ethernet son: `eth0`, `eth1`, `eth2`...

En nuestras computadoras ficticias solo tenemos una tarjeta que siempre se llamará `eth0`, y que vamos a configurar por ejemplo en la computadora `almacen`:

```
ifconfig eth0 192.168.1.4
```

Este interfáz además nos provee la ruta por donde enviamos información destinado para la red local, tenemos que establecer esta información, con la utilidad del sistema `route`:

```
route add -net 192.168.1.0 netmask 255.255.255.0 eth0
```

Como detalle notese, que al configurar el número IP de un interfáz de red el núcleo automáticamente establece una ruta hacia él, en otras sistemas habría que agregar esta ruta manualmente: "`route add 192.168.1.4 eth0`".

También este pasó tiene que establecerse permanentemente en el inicio del sistema, que en algunas distribuciones se hace con la misma herramienta gráfica al momento de declarar todos los datos del interfáz a "agregar" al sistema. En Debian Gnu/Linux se edita el archivo `/etc/network/interfaces`:

```
auto eth0
iface eth0 inet static
    address 192.168.1.4
    netmask 255.255.255.0
```

Una vez establecida esta entrada (una entrada para el dispositivo loopback ya debe existir), se puede configurar y desconfigurar ruta y número IP de la tarjeta Ethernet con los comandos: `if-up eth0`, `if-down eth0`. La línea "auto" en conjunto con los archivos de inicio garantiza, que este dispositivo se configura y desconfigura "automáticaente" al iniciarse y al apagarse el sistema operativo.

Este proceso no esta muy estandardizado entre diferentes distribuciones, y el sistema RedHat, por ejemplo, tiene comandos parecidos, pero que no son compatible y se configuran de forma diferente al esquema que utiliza Debian.

### Gateway

Si suponemos, que en la computadora de la secretaria hay un faxmodem el cual está configurado para conectarse al Internet, podemos usar esta computadora como enlace para el resto de las computadoras en la red. Esto requiere la configuración de la computadora `secre` como "Masquerading Firewall", un procedimiento que no vamos a extender aquí.

Para las otras computadoras en la red basta "saber", que `secre` es el "gateway", la ruta para tráfico de red que no está destinado a la red local. Para estos fines hay un "default" rule, una regla por defecto que se puede aplicar en la tabla de enrutamiento con el comando:

```
route add default gw 192.168.1.1 eth0
```

El parámetro "gw" indica, que los paquetes IP no son enviado a 192.168.1.1, sino a través de este interfáz. El parámetro "default" indica, que esta regla aplica para todos los paquetes en los cuales no aplica otra regla, o sea, que esta regla se agrega al final de la tabla de enrutamiento, y que no filtra el destino del paquete de ningúna manera. El parámetro "eth0" en realidad es superfluido, ya que através de las otras reglas es deducible, mediante cual interfáz de red de la computadora está accesible el dispositivo gateway. En vez de especificar el número IP del gateway puede usarse su nombre, pero

si el servidor de nombres no está funcionando el comando fallaría, peor, si asignamos a `secre` otro número IP, para usar una computadora dedicada como gateway, tendríamos que cambiar la configuración de todas las computadoras.

En el archivo de configuración de los interfaces de Debian agregamos una línea:

```
auto eth0
iface eth0 inet static
    address 192.168.1.4
    netmask 255.255.255.0
    gateway 192.168.1.1
```

En otros sistemas recurrimos a los programas gráficos de configuración de la red, o sus mecanismos respectivos de configuración.

### Nameserver

Aún en redes tan pequeños como es `cigarros.com` es recomendable instalar un nameserver, un servidor de nombres de dominio. La ventaja es, que al cambiar el número IP de una computadora o agregar una computadora nueva a la red solo hay que cambiarlo en esta misma y en el servidor de dominio, para las otras computadoras el cambio está disponible a través del servicio de nombres. Otra ventaja es, que el servidor de nombres puede realizar todas las solicitudes de nombres a una red remota y memorizar las respuestas, si nuestra minired tiene un enlace hacia el Internet (caching nameserver). De esta manera, si se repite la solicitud (un caso muy común) la respuesta se da a través del servidor de nombres local y no se genera tráfico en el enlace remoto. También la configuración de las computadoras es más sencilla, ya que no hay que informarles sobre servidores remotos de nombres, lo que se hace complicado cuando el enlace es intermitente.

El archivo `/etc/hosts` de todas las computadoras se reduce al interfáz loopback y al interfáz Ethernet, por ejemplo enel caso de `jofer`:

```
127.0.0.1 localhost
192.168.1.5 jofer.cigarros.com jofer
```

Pero se requiere la configuración del archivo `/etc/resolv.com`:

```
domain cigarros.com
nameserver 192.168.1.1
```

Aquí asumimos, que el servidor DNS está instalado en la computadora `secre`.

La configuración del servidor de nombres está fuera del marco del libro, pero esencialmente hay que crear dos archivos, uno para la resolución de nombres en númerios IP y uno para la resolución “inversa”, con listas como:

```
secre A 192.168.1.1.
jefe A 192.168.1.2.
venta A 192.168.1.3.
```

y

```
1. A secre
2. A jefe
3. A venta
```

# 9. Impresión

En unix existen dos elementos clave para la impresión: el unix printer spooler lpd (line printer daemon), y el uso extenso del formato Postscript para el formateo del contenido de impresión.

El printer spooler es un sistema o colección de programas que permiten remitir trabajos de impresión a uno computadora, inclusive remota, mantenerla en una cola de espera y enviarla a la impresora especificada, sin intervención del/de la usuario/a. De esta manera se pueden enviar varios trabajos de impresión al mismo tiempo, sin que hay colisión, ya que el sistema secuencializa su ejecución, y además se pueden ejecutar los trabajos con una prioridad baja, solo usando recursos del sistema, cuando este esté desocupado y así no se bloquean procesos interactivos de usuarios.

Existen dos implementaciones con diferentes comandos: los Unix de la tradición System V utilizan "lp/lpstat" como comando de remisión y de supervisión de trabajos de impresión, BSD-Unix utiliza "lpr/lpq" para este mismo efecto.

Desde la primera implementación se han sofisticado los programas y se han hecho más compatibles. CUPS (Common unified printing system) es una iniciativa que trata de crear un nuevo estándar para el control de trabajos de impresión, y que integra el estílo BSD y System V como subelementos.

El printer spooler no tiene (casi) ningún conocimiento ni influencia en el formato o la calidad de la impresión. Por un lado porque viene de un tiempo donde una impresora solo podía imprimir tipos de letras fijos, y no había (prácticamente) formas de imprimir información gráfica. Por otro lado es consistente con la filosofía Unix, de que cada programa solo realiza una tarea.

Mientras en MS-Windows cada aplicación tiene que contener el convertidor de impresión de su formato de archivo al formato de impresión de una impresora específica, en Unix es el sistema que se encarga de convertir los diferentes formatos de archivos en su forma imprimible. Ambos enfoques tienen sus ventajas y desventajas, en Unix es efectivamente la falta de un estándar suficientemente reconocido e implementado, que la impresión funcione bajo cualquier circunstancias.

El lenguaje de programación gráfica Postscript ha llegado a llenar a cierta medida esta deficiencia. Practicamente todos los programas en unix pueden convertir su formato interno al formato Postscript, o existen programas auxiliares que pueden realizar esta tarea, y con el programa Ghostscript, se puede convertir el formato Postscript en virtualmente cualquier formato de impresión que existe en el planeta. De esta manera, un sistema Unix puede combinar lo mejor de ambos mundos, siempre y cuando todos los componentes están en su lugar.

En este capítulo se presentará implementaciónes concretas del sistema de impresión y su configuración. Los detalles técnicos de la implementación pueden leerse en la literatura

correspondiente, que es muy extensa. La lectura puede comenzar con la documentación en línea de los programas LprNg y CUPSys, que es muy buena y ámplia.

## 9.1. Interfáz programático

A continuación se presentan los comandos usados comúnmente usados por el/la usuario/a para imprimir un archivo con una descripción corta en la comparación entre BSD y System V.

| Función | BSD | System V |
|---|---|---|
| imprimir un archivo | lpr *archivo* | lp *archivo* |
| imprimir de un pipe o stdin | lpr - | lp - |
| imprimir en una impresora específica | lpr -P*impresora archivo* | lp -*d impresora archivo* |
| revisar la cola de impresión | lpq -P*impresora* | lpstat -d *impresora* |
| cancelar un trabajo en cola | lprm -P*impresora trabajo* | cancel -d *impresora trabajo* |

Nótese, que al crear un trabajo de impresión (con lpr o lp), se retorna un *número de trabajo* (Print Job Number), en la tabla indicado con *trabajo*. En el uso diario pasa muchas veces, que se manda un archivo a la cola de impresión, y después se quiera cancelar. En este caso se revisa la cola con "lpq" o "lpstat para" obtener el número de trabajo, en caso (común) que no se conoce o no se ha memorizado al crear el trabajo. Este número se utiliza para cancelar el trabajo con "lprm" o con "cancel".

Para enviar un trabajo a la cola local es suficiente especificar cualquiera de los nombres de la cola u omitirlo completamente en caso se trata de la impresora por defecto "lp". Si queremos enviar un trabajo a una impresora remota podemos especificar su nombre concatenando el nombre de la cola remota con el de la computadora remota, por ejemplo: lp@venta.cigarros.com. En System V se puede especificar la computadora remota (host) con el parámetro "-h *computadora*".

No existe, ni es necesario una función "especial" del sistema operativo para la impresión, cada programa que quiera imprimir "algo" puede hacerlo creando un sub-proceso en el cual se ejecuta "lp -" o "lpr -", y redireccionar la información a imprimir al stdin de este sub-proceso. Esto es válido también para lenguajes scripting o programas shell. Obviamente también pueden crearse archivos temporales, usándose como argumentos para lp o lpr y que se eliminan a continuación, una técnica que puede proveer facilidad multiplataformas con sistemas operativos más pobres.

## 9.2. printcap

El archivo `/etc/printcap` es la clave de la configuración de las colas de impresión en Unix, independientemente del sistema lpd, sea BSD o System V. El nombre se derive de "printer capabilities" o capacidades de impresoras. Es un homólogo de la base de datos

termcap, que tiene una estructura parecida, pero está sustituida en la actualidad por un sistema más flexible y poderoso llamado terminfo.

/etc/printcap está compuesto por entradas, uno por línea, que describen cada uno una cola de impresión, con una gran variedad de parámetros posibles. Efectivamente se reestructuran las líneas con el símbolo de continuación de línea "\" para facilitar mejor legibilidad para los seres humanos.

Cada cola de impresión tiene uno o varios nombres, que conforman la primera entrada en la línea. Si son más que un nombre se separan por el símbolo "|". A continuación de los nombres se anexan campos de parámetros, separados entre ellos por el separador tradicional ":", que tiene la forma: `et=texto` o `et#número`, donde et es una etiqueta de dos letras que designa el parámetro a describir. a continuación presentamos dos entradas printcap mínimales, una para una impresora local y una para una cola de impresión en otra computadora - local y remote printcap entry:

```
lp|localprinter|impresora_local|Epson_LX300:\
    :lp=/dev/lp0:\
    :sd=/var/spool/lpd/lp:\
    :mx#0:\
    :sh:
```

Tradicionalmente la cola de impresión por defecto se llama lp, quiere decir, si ejecutamos el comando "lpr *archivo*", sin especificar la cola de impresión se imprimirá en la impresora con nombre lp. No hay necesidad de ponerle varios nombres a la cola de impresión, pero es recomendable darle un nombre corto "unix" de máximamente tres letras, y otro nombre que describe el modelo de la impresora. De esta manera se puede cambiar por ejemplo la impresora "lp" por un modelo diferente, darse cuenta como administrador del sistema, pero mantener el interfáz de usuario (el primer nombre) igual para los usuarios, y por otro lado, como usuario se puede seleccionar la impresora por el nombre del modelo, que es más palpable, teniendo el objeto físico (la impresora) en frente, que un nombre abstracto de la cola.

El símbolo "\" tiene que ser el último símbolo antes del símbolo de nueva línea (\n). Entonces el contenido de la siguiente línea será concatenado a la actual, omitiendo le espacio blanco al comienzo de la línea, por lo que se permite hacer la sangría para estructurar mejor la apariencia. Lectores atentas/os se darán cuenta, que todas las entradas entonces se presentan así: "`lp|...::lp=/dev/lp0::sd=...`", con dos doble puntos. Esto no consiste un problema dado que campos vacíos (el "no"-espacio entre dos colones) son ignorados en /etc/printcap.

**lp=** designa el dispositivo físico - el puerto paralelo 1 de la computadora en este caso, donde está conectada la impresora (lineprinter - port).

**sd=** designa el directorio donde se almacenan los trabajos: la cola de impresión o "spool directory".

**mx#** es el tamaño máximo de un archivo que se imprime en una impresora, utilizando el valor cero significa no imponer una restricción al tamaño.

**sh** es un parámetro sin valor y significa, que se crea una copia del archivo en el directorio de impresión, que va siendo eliminando al terminar el trabajo. Tradicionalmente pueden haber otros enfoques:

- el archivo no se copia, sino se crea un enlace simbólico en el directorio cola. Esto es mala idea, cuando el archivo cambia o es borrado durante la impresión o espera del trabajo, aunque puede ser útil cuando los archivos a imprimir son grandes y muchos y el espacio en el disco duro es poco.
- el archivo original va siendo eliminado al terminar la impresión. Sin comentario.

```
rlp|Remote printer entry|impresora_remota|LaserWriter:\
    :lp=:\
    :rm=venta.cigarros.com:\
    :rp=lp:\
    :sd=/var/spool/lpd/rlp:\
    :mx#0:\
    :sh:
```

Esta entrada tiene definido, que el puerto de impresión es vacío, para designar que la impresión es remota. Se requiere saber el nombre de la cola en la computadora remota, y por supuesto el nombre de la computadora, que es definido en las entradas "rm" - remote host, o computadora remota, y "rp" - remote printer, o impresora remota. En nuestro caso, "rlp" refiere la impresión a la cola de impresión por defecto en la computadora venta.cigarros.com, y suponemos, que este sera una impresora del modelo "LaserWriter".

Consta, que siempre requerimos un directorio de cola, ya que el trabajo se guarda localmente, mientras es transmitido a través de la red. Si la computadora remota no es accesible, la transmisión se reintenta cada vez en cuando.

De particular interés son entradas no mostradas aquí, que permiten "filtrar" el archivo de impresión. Hay filtros de entrada (if=), salida (of=) y para la contabilidad (af=), además existen filtros para diferentes formatos de archivo, que hoy día ya no tienen mucha importancia. Los filtros designan la vía de acceso a un programa, que es llamado en un determinado momento del procesamiento del trabajo de impresión, recibe alguna información del sistema de impresión mismo a través de variables de entorno y a través de parámetros de línea de comando, y que tiene que leer en su stdin el archivo de impresión, y escribir en su stdout la transformación de esta información.

El archivo de entrada es usado, por los sistemas de impresión genérica, para dedectar el formato de archivo, y crear la fórma óptima de impresión para la impresora específica. Se puede decir, que en los sistemas "modernos", el input-filter es el driver de la impresora, pero combinado con un convertidor "intelligente" de formatos.

El accounting-filter (af) es usado para llevar la contabilidad de hojas impresas en esta cola. La información que recibe en cada trabajo le permite crear una estadística por computadora, cuenta y período de tiempo, y/o crear una facturación de costos de impresión por cuenta, que puede ser de interés en una institución o empresa.

## 9.3. LprNg e impresión remota

LprNg es una re-implementación del sistema BSD de impresión, con una grán cantidad de mejoras y sofisticaciones, especialmente para trabajo en redes y con múltiples usuari@s e impresoras.

En su instalación básica sustituye sin más complicaciónes al sistema Lpr de la distribución BSD, y puede configurarse a través del archivo printcap como descrito anteriormente. LprNg utiliza un segundo archivo de configuración (opcional), donde se describe los derechos de acceso a las colas de impresión. Se puede asignar derechos de enviar trabajos, revisar la cola, reordenarla, borrar un trabajo de la cola en dependencia de usuario, nombre de computadora y número IP de una manera muy flexible.

LprNg en sus versiones recientes también provee herramientas gráficas para su configuración, que en su apariencia es parecido al Printtol de la distribución RedHat.

LprNg es una implementación muy madura, que satisface igual las necesidades de una instalación individual en una computadora personal, como una aplicación profesional con necesidad de soporte para el manejo de redes grandes y complejos y con posibilidad y soporte directo para la contabilidad de impresión.

## 9.4. Cupsys

Es una implementación completamente renovada, surgiendo de las opciones avanzadas que se presentan por la inovación tecnológica en las impresoras.

Mientras lpd mira a la impresora como un equipo de solo escritura, en la actualidad hay computadoras que soportan una comunicación bidireccional, sea para la configuració óptima del uso de tinta o toner, para notificar datos de estadística de las impresiones y para comunicar a la computadora modelo, versión y capacidades.

Cups además está diseñado para poder integrar impresión en red desde múltiples plataformas y protocolos de red e implementa en una impresora colas de impresión para redes Microsoft, Novell, Unix System V y BSD.

La configuración de CUPS se realiza a través de un navegador Web, dirigiendose comúnmente al puerto TCP/IP número 631. Por ejemplo para configurar la(s) impresora(s) local(es) de una computadora se dirige cualquier navegador al URL `http://localhost:631/`.

## 9.5. Ghostscript

El lenguaje Postscript fue desarrollado por la empresa Adobe, como formato de representación gráfica. La idea consiste en que una imágen, compuesta por figuras geométricas y por texto en tipos de letras y tamaños arbitrario puede ser representado de forma descriptiva. La formaliciación de esta descripción forma un lenguaje de programación. Supongamonos un cuadro que consiste de un circulo negro con fondo azul-celeste en la parte izqierda superior, con un rectangulo amarillo de marco negro interseccionando en

la parte inferior derecha del circulo, y el texto "La Femmé" en letra Roman, al tamaño de dos pulgadas de altura en el centro de la parte inferior del cuadro.

Entonces a esta descripción le faltan solo poco detalles, como las medidas del líenzo, y la posición exacta de los objetos, para reproducirlo exáctamente. Es óbvio, que podemos dar las instrucciones en diferente orden y obtener el mismo resultado.

Postscript trabaja con un lienzo imaginario "canvas", en el cual se supone una cuadricula en points (1/72 pulgadas) con referencia (x,y)=(0,0) en el punto inferior izquierda, en el cual se "dibujan" figuras geométricas, y letras, u objetos como archivos gráficos de mapas de bits, u objetos más complejos definidos en el mismo lenguaje Postscript, que pueden ser posicionados, virados, cizallados, etc.

De otro punto de vista se puede decir, que Postscript es una bibliotéca de funciones gráficas, aglomerado con un lenguaje completo de programación. El lenguaje de programación es derivado del lenguaje Forth, con notación polaca inversa. Una propiedad muy específica de este formato es, que solamente utiliza caracteres ASCII y por lo tanto es portable entre cualquier sistema de comunicación o procesamiento de datos.

Total, el conocimiento de estas maravillas tenemos, porque la empresa Adobe abrió al público en general la definición completa de Postscript, que existe en dos niveles: Postscript Level 1 y Postscript Level 2. Hace poco tiempo, después de un largo tiempo de uso y divulgación del lenguaje Postscript, Adobe amplió el lenguaje con habilidades de hyperenlaces, y con operadores que permiten la compresión/decompresión interna de archivos[1], ahora denominado Portable Document Format - PDF.

En el área de las impresoras profesionales es muy común, que un equipo tenga incorporado el lenguaje Postscript, lo que significa, que en vez de utilizar un gestor específico para la impresora, se le manda directamente comandos Postscript que describen el diseño de la página, y la impresora la convierte en su correspondiente imagen. Esto se encuentra muy frecuentemente en impresoras Laser.

A Peter Deutsch le debemos un interpretador de este lenguaje que es distribuido mediante la Licencia Pública, llamandose Ghostscript. Un interpretador del lenguaje Postscript es un programa (o filtro en términología Unix), que lee un archivo Postscript y lo convierte en ... otro formato. Ghostscript puede generar los formatos y comandos de virtualmente todas las impresoras del planeta, que son capazes de crear no solo texto sino también imágenes, y además puede convertir el formato Postscript en otros formatos gráficos. Podemos diferenciar varios tipos de "lenguajes gráficos" y "lenguajes de impresoras". Formatos de imágenes pueden ser en mápas de bits, que es una representación simple y requiere mucho volumen, o descriptivo, o también con métodos de compresión, donde se omiten detalles, que para el ojo humano no tienen mucha relevancia. En las impresoras hay diferentes formatos, que corresponden a la técnica de impresión. Impresoras matriziales de impacto trazan varias líneas de una descomposición matrizial en un barrido; impresoras laser imprimen justo una sola línea, pero pueden cargar a veces una páginas entera en su memoria, impresoras a color tienen una intercalación compleja de válvulas diminutas con diferentes colores cuya composición realiza el color de la imágen.

[1] Lo que a) rompe con la representación estríctamente ASCII b) introduce problemas legales, porque se utilizan algorítmos propietarios y también encripción que tiene estatus especial en algunos países

Ya que la distribución del programa incluye el código fuente con todos los gestores, pueden compilarse interpretadores Ghostscript diversos, que incluyen los gestores que sean, e inclusive es realtivamente fácil crear e integrar nuevos gestores. Aunque sería posible, crear para cada impresora diferente un propio ejecutable de Ghostscript es común, compilar un conjunto de gestores para dispositivos desde la tarjeta gráfica en modo VGA, para X-Windows, para diferentes formátos gráficos y hasta un rango ámplio de impresoras comunes en un solo binario en las distribuciones de Linux. De esta manera se puede utilizar un solo programa para convertir de Postscript y PDF en todos los formatos probablemente necesarios. Incluso, Ghostscript puede crear como salida de nuevo un archivo Postscript o PDF.

Esto último es muy útil en la versión Ghostscript para Windows, porque puede usarse para crear archivos en formato PDF.

Una línea de comando común para convertir un archivo Postscript (con nombre "printfile.ps") en un formato de impresora puede ser la siguiente:

```
gs -sOutputFile=printfile.epson -sDEVICE=epson -dNOPAUSE -q printfile.ps
```

**gs** es el nombre del programa Ghostscript

**-sOutputFile** define, que el resultado de la conversión debe escribirse en el archivo printfile.epson

**-sDEVICE** define el dispositivo para el cual debe realizarse la conversión, o en corto: el formato de salida. En este caso se convierte el archivo de entrado en un archivo que puede imprimirse directamente por una impresora matricial de la empresa Epson de 24 pines.

**-dNOPAUSE** instruye a Ghostscript, que no pide confirmación después de convertir una página, lo que es el comportamiento normal cuando se utiliza Ghostscript como visualizador, p.ej. en X-Windows.

**-q** (quiet=silencioso) reduce la cantidad de información de progreso impresa en la salida stdout

Para verificar, cuáles formatos de salida son soportados por una versió específica de Ghostscript se puede invocar el comando "gs" con el parámetro "-help" o "-h" solamente. En este caso Ghostscript solo imprime toda su configuración en la salida estándar y termina.

Aunque con el comando "`gs archivo.ps`" se puede visualizar comúnmente un archivo Postscript, sea en X-windows o sea en modo gráfico de la consola, es más práctico utilizar un visualizador, dedicado. Ghostview fue un programa diseñado como "manejador" (frontend) para Ghostscript, y fue a continuación sustituido por "gv", ambos programas del ambiente X-windows, que proveen un ambiente fácil de usar por el/la usuario para visualización e impresión selectiva de un archivo Postscript.

Regresando al tema de la impresión: se puede establecer un filtro de impresión en `/etc/printcap`, que converte cualquier impresora en una impresora Postscript de la siguiente manera:

```
if=gs -sOutputFile=- -sDEVICE=epson -dNOPAUSE -q -- -
```

donde por supuesto "epson" se sustituye por el modelo de la impresora en cuestión. el guíon "-" al final de la línea significa leer el archivo de entrada desde stdin, y el guíon en "-sOutputFile=" significa escribir la salida de Ghostscript en stdout. El doble guíon es necesario, para que Ghostscript interprete el guíon final no como opción de línea de comando.

Postscript, a pesar de preservar el formáto gráfico a través de diferentes plataformas, también cuenta con una grán cantidad de utilidades, que se sirven de que se trata de un lenguaje de programación. Se pueden hacer varios tipos de transformaciones a un archivo existente, al "procesarlos" con "programas Postscript" pre-cargados al interpretador, algunos ejemplos que todos son programas del paquete "psutils" de Angus Duggan:

**psnup** convierte un archivo Postscript que consiste de páginas consecutivas en uno, donde "n" (n=1,2,...) páginas están reproducidas en cada hoja consecutiva de forma reducida y reorientada.

**psbook** convierte un archivo Postscript de tál manera, que al imprimirlo puede ser doblado y engrapado como un libro o folleto

**psmerge** junta varios archivos diferentes Postscript en un solo

**psselect** permite extraer determinadas páginas de un archivo Postscript.

Por fín falta mencionar, que a través de convenciones de programación en Postscript se distingue entre diferentes "formatos". Para la reproducción de archivos Postscript en papel, donde el énfasis es la separación en páginas existen los "Adobe Document Structuring Conventions" -DSC, comentarios insertados en el archivo, que describen los limites de páginas, su númeración etc. Programas pueden extraer estos "comentarios" y utilizarlos para el manejo del documento. Por ejemplo, el visualizador muestra el número de páginas y permite buscar una página específica a través de ellos.

Encapsulated Postscript o EPS, es un formato Postscript para dibujar una imagen, que puede ser redimensionado, insertado en otro archivo como objeto gráfico, etc. EPSI o Encapsulated Postscript Interchange Format incrusta una imágen reducida un "thumbnail" dentro del archivo mismo, que puede ser usado por visualizadores para una vista rápida del contenido. A la hora de impresión se utiliza el archivo original y se oprima el "thumbnail".

## 9.6. Magicfilter

La solución propuesta por el capítulo anterior al problema de la impresión de archivos gráficos con el sistema lpd es, que todos los programas que quieren imprimir tienen que

convertir su salida al formato Postscript por un lado, y por otro lado que todas las impresoras sean impresoras Postscript, o sean "convertido" en tal mediante Ghostscript como filtro de entrada. Nótese, que esto significa, que hasta un simple texto ASCII tiene que convertirse en Postscript antes de imprimirse. Existen un sinnúmero de utilidades para la transformación de archivos cualquieras en este formato, unos pocos jemplos son:

**a2ps** "All to Postscript" - todo a Postscript

**enscript** convierte archivos ASCII en Postscript y le agrega formato, como línea título con la fecha de impresión, el nómbre de archivo, etc., númera las páginas, y puede imprimir también varias páginas en un sola hoja. Además, enscript provee un mecanismo para resaltar el síntaxis de muchos diferentes lenguajes de programación (language markup), lo que facilita la lectura de programas de computación significantemente.

**pnm2ps** convierte archivos pnm (imágenes de mapas de bits) arbitrarios en formato Postscript, y eventualmente los hace calzar dentro de los márgenes de una página.

Esta propuesta transfiere el problema "Cómo imprimir un archivo X en la impresora Y" solamente de un nivel a otro: "Cómo convierto el archivo X en formato Postscript". No debe ser un problema concernante para el/la operadora de la computadora, sino para la computadora mismo, y eso es, para que se invento los filtros mágicos.

El concepto de estos es, que se le agrega "inteligencia" al filtro, para que detecte el tipo de archivo autónomamente, e invoce la conversión correspondiente a la impresora. Para eso no es necesario, que el archivo siempre se convierte en Postscript, por ejemplo si se trata de un archivo ASCII.

El programa "magicfilter" es una implementación bastante madurada de este concepto. Magicfilter utiliza un archivo de "configuración" para cada tipo de impresora, donde una línea describe una conversión. Una línea comienza con el "magic" el identificador del tipo de archivo, después indica que forma de transformación se pretende hacer: pasarlo directo, pasar lo por un filtro, pasar lo por un programa, o pasarlo por un filtro/programa y después volver a evaluarlo. Esta última opción introduce recursión en el procesamiento y le da su poder a magifilter. La última parte de la línea indica la línea de comando a invocar.

Lo interesante de Magicfilter es, que el/la usuario/a no tiene que escribir estos archivos de "configuración", ya que Magicfilter incluye un programa de configuración, que analiza, cuales convertidores existen en la computadora y compone el archivo correspondiente. Como gestor de impresora, o sea el filtro finál para archivos gráficos se utiliza Ghostscript. La utilidad de configuración crear el archivo `/etc/printap` y se rehusa a ejecutar si este ya existe, por ejemplo cuando printcap fue creado por LprNg u otra utilidad. En el caso, que la configuración de printcap tiene que conservarse se procede moviendo el archivo original a un lugar "seguro", creando un archivo printcap con "magicfilterconfig", donde se selecciona el tipo de impresora, y después se incorpora las líneas relevantes de este archivo en el original. Finalmente se vuelve a copiar el archivo printcap original (modificado) a su lugar.

Magicfilter puede instalarse directamente de su código fuente sin mucho problema. La distribución Debian Gnu/Linux provee un paquete preconfigurado de Magicfilter, con gestores para una grán variedad de impresoras comúnes.

## 9.7. Redhat-Printtool

La distribución RedHat, y a mi parecer también la distribución de Suse utilizan otra técnica, que también tiene tradición en Unix. Dentro del directorio de cola de lpd se instalan los scripts y filtros correspondientes para la impresora. Ellos utilizan un sistema de filtraje mágico, que se basa únicamente en el interpretador de comandos bourne shell.

La configuración y adaptación de los filtros para una nueva impresora, así como la instalación de los scripts (porque son necesarios varios) en su lugar correcto es una tarea no trivial. Sin embargo existe una aplicación gráfica, llamada "`printtool`", que está muy bién diseñada, y permite al/a la usuario/a una configuración y administración extremadamente sencilla. Printtool también puede tratar con impresoras en red, y no solo de colas lpd de Unix, sino también para colas de impresión Novell y Redes Microsoft.

De nuevo, igual como en la configuración de X-Window, se saca provecho de una base de datos de configuración, en este caso de impresoras, bien diseñada y madurada.

Printtool se basa en el sistema lpr de BSD y obviamente no existe un control de acceso como lo provee LprNg o CUPS. En redes grandes, donde la seguridad o privacidad es un tema que hay que tomar en cuenta tiene que valorarse esta limitación.

# 10. Redes Heterogeneas

Una red consiste en su estructura física y la tecnología de interconexión de varias computadora para el intercambio de datos entre ellos, y en la forma, cómo estos datos están estructurados e interpretados - el protocolo de red. Igual como entre las personas, el lenguaje en que hablamos puede ser diferente, pero transmitimos contenidos identicos a pesar de ello, "hablan" las redes de diferente procedencia diferentes "idiomas", pero se trata de las mismas tareas a realizar. Para extender la analogía más todavía, si en un cuarto hay varias persionas que hablan diferentes idiomas, pueden comunicarse los que hablan entre si el mismo, sin interferir en las comunicaciones de los demás. En este escenario, Linux tiene todas las facilidades de un intérprete de varios idiomas, a través de varios paquete de software que permiten que "hable" diferentes "idiomas" - protocolos de red, eso sí, con las mismas restricciones de un intérprete humano. Hay ciertos términos o contextos que estos paquetes (todavía) no manejan, esto por un lado, pero por otro lado también hay conceptos fuera del simple idioma, que en un ambiente tienen sentido y en otro no tienen correspondencia. Si un grupo de zapateros ingleses se encuentra con un grupo de sastres alemanes podrán compartir aspectos comunes de su profesión, sin embargo sus habilidades profesionales difieren, y esta discrepancia no puede ser superado por el intérprete.

En este capítulo verémos cuatro diferentes protocolo de redes comunes, que podemos emplear en Linux. Para salír de la metáfora hay que destacar que para una computadora que es "nativa" de un determinado protocolo de red, de todas las computadoras físicamente conectadas en red solamente las computadoras que utilizan el mismo protocolo son "visible". Con el término "visible" expresamos la posibilidad que una computadora puede ofrecer o solicitar un servicio a otra. Estamos hablando de un modelo Servidor-Cliente, en oposición a un modelo Maestro-Esclavo. En todas las redes que vamos a analizar, cada computadora en principio puede ser Servidor de servicios a una o varias otras, y Cliente de otras computadoras en el mismo momento.

Los servicios básicos de la computación en redes son: Servicio de Archivos y Servicio de Impresión.

Nos vamos a concentrar en estos dos, ya que son comunes y parecidos a todas las redes. El servicio de mensajes también existe en todas, sin embargo allí ya empiezan diferencias en cuanto a la pregunta ¿qué es un mensaje?

El protocolo tal vez más elaborado y flexible es TCP/IP, el protocolo utilizado en Internet y cual es el protocolo "nativo" de Unix y por lo tanto también de Linux. TCP/IP se presta a la creación de nuevos protocolos. En Linux podemos ver un espectro de ello en el archivo `/etc/services` y a continuación vamos a hacer una comparación de los servicios oferidos por los protocolos de red en discusión por este documento[1].

[1] Esta comparación necesariamente queda corta, mientras pueda recolectar más información sobre los

| Servicio | TCP/IP | MS-Net | Novell | AppleTalk |
|---|---|---|---|---|
| Impresión | lpr | | | |
| | cups | | | |
| | samba, smbfs | si | | |
| | mars, núcleo | si | si | |
| | netatalk | | | si |
| Archivos | nfs | | | |
| | coda | | | |
| | samba | si | | |
| | mars, ncpfs | si | si | |
| | netatalk | | | si |
| Mensajes | winpopup | si | | |
| | talk | | | |
| | ? | | si | |
| Ejecución remota | telnet | | | |
| | rsh | | | |
| | ssh | | | |
| Correo electrónico | smtp | | | |
| | pop | | | |
| | imap | | | |
| | | ? | | |
| | | | | |
| | | | | |

Por supuesto que todos los sistemas operativos cuyo protocolo de red NO es TCP/IP tienen también implementaciones de este protocolo, e implementan programas Clientes y Servidores para estos servicios en ello, pero nuestro enfoque en este capítulo es: cómo utilizar y configurar los servicios "ajenos" que ofrece Línux a los otros.

## 10.1. Redes Microsoft - Samba-Suite

El protocolo SMB - Session Message Block es utilizado en las redes implementados por los sistemas operativos de Microsoft, a partir de Windows 3.11. Existen dos implementaciones de ello para MS-Dos, que pueden inclusive bajarse gratuitamente de los sitios de esta empresa, uno de ellos llamado LanManager.

Durante la evolución de los sistemas operativos de Microsoft se han hecho una variedad de cambios e variaciones en los diferentes implementaciones de programas clientes y servidores, lo que hace más difícil adaptar y configurar un programa para atender a este tipo de redes, ya que según las versiones del sistema operativo de las diferentes computadoras que participan en la red hay diferentes requerimientos y problemas que atender.

La ídea original de las redes Microsoft - facilitar el trabajo en grupo - implementa un

protocolos no-TCP/IP.

control de acceso muy leve, llamado "SHARE-Level access control" y fue complementado en versiones posteriores con "USER-Level access control". Lo primero se refiere, a los objetos compartidos ("shared" en inglés) - directorios e impresoras. Para que un cliente gane acceso a un SHARE, el/la usuario/a tiene que introducir una cláve relacionada con este objeto - una mala idea, considerando que esto significa la (relativa) publicidad de una clave de acceso. A partir de Windows NT y el sistema de archivos correspondiente - ntfs, se introduce el concepto de la propiedad de objetos en relación con usuarias/os del sistema. El acceso a un objeto se concede entonces, si a) el/la usuario/a se ha anteriormente autentificado en la red b) el/la usuario/a tiene derecho de uso del objeto. La instancia responsable para la autentificación es el PDC el "Primary Domain Controller", al cual otras computadoras en la red solicitan información sobre el estátus de autenticidad de un/a usuario/a en la red cuando lo requieren.

Un concepto fundamental en las redes es el "Browsing", la "visibilidad" de computadoras y servicios. Redes Microsoft casi no requieren una planificación de su configuración. Las computadoras intercambian periódicamente información sobre las computadoras disponibles y sus servicios. De esta manera una computadora puede encenderse y apagarse en cualquier momento, y dentro de algún tiempo (0 a 15 minutos) todos los potenciales Clientes están enterado de presencia o ausencia de un cierto servicio. La identificación de servicios en la red se representa en tres datos:

**Grupo de Trabajo** que corresponde a algo parecido como el dominio en TCP/IP. Es un nombre textual, que es parte de la configuración de la red. Todas las computadoras en un "Workgroup" pueden "verse" mutuamente.

**Nombre de Computadora** Es un nombre textual. No existen números IP adicionales. Ya que se consideran redes locales no existe el concepto del gateway - de que una computadora puede estar presente en varias redes interconectados (por lo menos en versiones "viejas" de SMB).

**Nombre del recurso compartido** designa el directorio o la cola de impresión que es disponible para otros/otras participantes en la red.

Mientras el grupo de trabajo solo aparece en la configuración o selección de "Subgrupos" ortogonales de computadoras, se requieren los otros dos componentes para formar un nombre completo de un recurso.

```
\\computadora\compartido
```

Es la notación que se utiliza en redes Microsoft para designar a los programas clientes el servicio a solicitar. Notese el símbolo prefijo "\\" para designar la computadora y "\" - el separador tradicional de MS-DOS para anexar el recurso.

Diferentes computadoras en la red pueden fungir como Browse-Server en la red. Entre ellos se elige de manera automatizada una de ellos que se denomina "Master Broswer", el cual sirve de base de datos para computadoras entrantes en la red. El Master Browser y el Domain Controller pueden ser la misma computadora y muchas veces lo suelen ser.

A la hora de que las redes Microsoft aumentan de tamaño se introduce el concepto de "Dominio", aunque de la documentación de Samba no se puede deducir con claridad su significado.

El paquete de Software que implementa programas Clientes y Servidores en Linux (Unix), se denomina "Samba" y fue percebido en primer lugar por Andrew Tridgell, que además nos regaló el maravilloso programa "rsync".

Samba implementa el protocolo SMB hasta en su expresión en Windows NT 4.0, con casí toda la funcionalidad, con la salvedad que utiliza (actualmente) a TCP/IP como protocolo de transporte. Esto significa, que las computadoras clientes un sistema operativo Microsoft (Windows) que participan en una red Microsoft con computadoras Linux tienen que instalar el protocolo TCP/IP para que pueden "ver" a la computadora Linux. Lógicamente tiene que configurarse este protocol de tal manera, que hay una conexión de red a nivel TCP/IP, quiere decir asignarles números IP's válidos, darles a conocer el servidor nombre en la configuración y/o editar el archivo LMHOSTS en el directorio Windows de la misma manera como /etc/hosts en las computadoras Linux (Unix).

La implementación del programa cliente SMB se llama "smbclient", una aplicación de línea de comando, para proveer los servicios SMB se sirve de dos programas que corren en modo "daemon" o que son lanzados por solicitud mediante `inetd`: "smbd" y "nmbd". El primero resuelve el transporte de información y el segundo realiza las funciones del "browsing".

### 10.1.1. Conectar Linux a un servidor SMB

Para utilizar los servicios compartidos de otras computadoras disponemos básicamente de `smbclient`, que incluye todos los servicios en una sola aplicación: browsing, impresión, acceso a archivos y envión de mensajes. Hay una variedad creciente de aplicaciones gráficas para facilitar el manejo de redes Microsoft, de los cuales la mayoría recurre internamente a llamadas al programa `smbclient`. Hay que notarse, que smbcliente permite la especificación del grupo de trabajo en cada llamada, por lo que obtenemos acceso a cualquier recurso SMB que es accesible mediante TCP/IP, independientemente de su organización en la Red Microsoft.

**TkSmb** es una aplicación Browser escrita en Tcl/Tk, que necesita, como punto de partida "conocer" una computadora. Usando el browsing list obtenida de esta, explora la red y obtiene todos los grupos de trabajo y sus servicios compartidos. TkSmb permite, con doble click, descargar archivos remotos en la computadora local.

**Gnomba** es una aplicación Browser del Gnome Desktop, que explora la red Microsoft y la presenta en forma gráfica, similar al "Ambiente de Red" en MS-Windows. Se puede "conectar" a directorios compartidos mediante programas externos a Gnomba, normalmente "smbmount" y "gmc", quiere decir al hacer doble click sobre un SHARE se abre una ventana del explorador de archivos, mediante el cual se puede copiar desde y hasta el directorio "conectado" (montado).

**smb2www** Es un script CGI, que se puede instalar en el servidor Web de una computadora Linux. Al visitar la página correspondiente smb2www presenta un ambiente gráfico similar al "Ambiente de Red" de MS-Windos, con todos los grupos de trabajo, computadoras y servicios conocidos en el navegador. De esta manera se puede entrar en un directorio compartido y al hacer doble click sobre un archivo (o directorio) bajarlo a la computadora local.

smb2www no necesariamente reside en la misma computadora que usa para acceder mediante smbclient a la red Microsoft. Debído a su carácter CGI, smb2www tiene que ser aplicado con cautela y solamente en redes locales que inhiben el acceso externo mediante un cortafuegos debídamente asegurado.

El suite (colección) Samba también provee dos programas, para montar un directorio remoto compartido en la computadora local, que se llaman `smbmount` y `smbumount` respectivamente. Estos pueden usarse como programas del espacio usuario, lo que segnifica que son independientes del núcleo. Núcleos Linux actuales ya proveen soporte para el acceso a directorios compartidos; el módulo de núcleo responsable se llama smbfs - Samba File System. En este caso los comandos del sistema `mount` y `umount` están previsto del sintáxis y la funcionalidad para acceder a directorios compartidos mediante SMB. Resumimos este comando en su aplicación en redes - que es idéntica a `smbmount/smbumount` en el caso de redes SMB.

```
mount /dev/hda1 /mnt -o ro
```

monta una partición de un disco duro en el directorio (punto de montaje) /mnt en modo de solo lectura. `/dev/hda1` es la vía de acceso a un "nodo especial", que es como una puerta de entrada a los gestores del núcleo para el disco duro. De allí detecta a) que se trata de un dispositivo de bloques y sus parámetros, b) analiza el sistema de archivos y aplica el gestor correspondiente - siempre y cuando existe en el sistema. La opción "`-t` *fs*" puede utilizarse para forzar un gestor para el sistema de archivos *fs* específico y así suprimir la autodetección.

```
mount jefe.cigarros.com:/usr/share/doc /usr/share/doc
```

Es un ejemplo como "ahorrase" copias duplicados de la documentación del sistema en una red mediante un servidor NFS (Network File System). La computadora `jefe.cigarros.com` "exporta" respectivamente "comparte" el directorio `/usr/share/doc`, y nuestra computadora local lo monta en el directorio equivalente. Notese la separación entre el servidor y el servicio brindado mediante el doble punto ":".

```
mount //jofer/listas /mnt
```

es equivalente a

```
mount \\\\jofer\\listas /mnt
```

Samba permite sustituir la "\" original de la red Microsoft por "/", lo que simplifica la notacíon, ya que en Unix la "\" es el caracter de "escape" para poder introducir símbolos especiales, y tiene que ser duplificado para recuperar su significado original - "\\" se convierte en "\". Vemos, que la notación distinguida permite a `mount` deducir el tipo de sistema de archivo a utilizar, en este caso smbfs.

Una particularidad del uso de transporte TCP/IP para el protocolo SMB es, que la computadora es accesible mediante la resolución de nombres tradicional DNS y su número IP, pero también mediante el protocolo NetBios desde su "nombre Netbios". Este no necesariamente tiene que ser el mismo y los programas de Samba lo toman en cuenta. (smb)mount acepta mediante la opción de línea de comando parametros para el gestor de sistema de archivos específico. smbfs soporta (entre otros) los siguientes parámetros:

**ip=*Nº-IP*** para especificar el númerio IP de la computadora a contactar

**netbiosname=*computadora*** para especificar el nombre netbios, necesario si difiere del nombre DNS

**workgroup=*Grupo-de-Trabajo*** para especificar el grupo de trabajo de la computadora servidor a contactar

**username=*Usuario*** Nombre de usuario para la autentificación

**password=*Cave*** especifica la clave de acceso, sea de usuario o para acceso de nivel de compartido

**guest** para indicar que debe de usarse acceso anónimo al servicio indicado.

### Cliente universal Samba

El programa `smbclient` es modelado según el cliente ftp. En vez de especificar un servidor ftp se especifica un servicio compartido en otra computadora en la línea de comandos y smbcliente presenta una línea de comandos, desde la cuál se pueden bajar archivos con el comando `get`, subir con el comando `put`, ver directorios y cambiar hacia ellos con `ls` y `cd`, etc. Hay un comando "`print`" que permite enviar un archivo local a una cola de impresión remota.

El hecho de que se puede ejecutar un comando desde la línea de comando de `smbclient` sín entrar en el modo interactivo permite la aplicación descrita por ejemplo en TkSmb, o para usar smbcliente como cola de impresión remota en `/etc/printcap`:

```
smblp|Samba Printer|LaserWriter:\
    :lp=/dev/null:\
    :sd=/var/spool/lpd/smblp:\
    :if=smbclient -c “print -“ //venta/lpr:\
    :mx#0:\
    :sh:
```

Daría acceso a la impresora Laser de nuestra red ejemplar a la que accedemos con una entrada remota en capítulo 8.5.6, siempre y cuando la computadora venta.cigarros.com tuviera instalado y configurado correspondiente el servidor Samba, pero también si venta.cigarros.com trabajaría en modo MS-Windows y tiene activado el compartir de la(s) impresora(s)[2].

smbclient tiene una gran cantidad de opciones de línea de comando, de los cuales discutimos algunos destacados:

**-M Computadora** Envia un mensaje Winpopup a la computadora índicada. Si está activada la recepción en aquella computadora el/la usuario recibe el mensaje.

**-L Computadora** obtiene la lista de servicios de la computadora indicada (nombre netbios). Para explorar una red esta opción es muy útil en combinación con:

**-I Nº-IP** que especifica el número IP de la computadora remota en vez del nombre netbios, que puede quedar vacio entonces.

**-W Grupo-de-Trabajo** especifica el grupo de trabajo en cual ubicarse

**-U Usuario** especifica, como cuál usuario autentificarse en la red. `smbclient` siembre pide la clave de acceso aunque no sea necesaria para el servicio a cual se acude. Esta funciónalidad se puede deshabilitar con

**-N** no pedir clave de acceso.

**-c commando** Especifica un comando de `smbclient` a ejecutar, sin entrar en modo interactivo.

### 10.1.2. Proveer Servicios SMB desde Linux

El servidor Samba consiste de los dos daemon-ios nmbd y smbd, que proveen el protocolo NetBios, y el protocolo SMB. La configuración permite una vasta cantidad de opciones para tratar con clientes de todos los niveles y variaciones del protocolo SMB. Dentro del marco de este Documento no se puede abarcar todos los aspectos y referimos a la amplia documentación disponible del mismo paquete samba, que no solamente provee información sobre el software, sino también sobre redes Microsoft, los sistemas operativos que los implementan y sus particularidades.

En vez de describir los diferentes problemas que tiene que resolver Samba para integrarse en estos, solo hacemos mención de algunos aspectos correspondientes:

**Nombres de máquinas** Ya hemos mencionado la doble "personalidad" que obtiene una computadora al presentarse en DNS-TCP/IP y en NetBios. En versiones más recientes del protocolo SMB Microsoft introduce el concepto "dominio" a la par del

[2] Esto solo es un ejemplo y no recomendable para el uso real, ya que en caso de una cola de impresión "real" de MS-Windows éste no provee ningúna transformación de contenido del archivo a imprimir a como lo haría p.ej. Magicfilter. Entonces tiene que realizarse el filtraje en la computdora local, lo que implica en el mejor de los casos una entrada en printcap con un mecanismo de "filtro mágico".

"grupo de trabajo", un término que a veces se utiliza como sinónimo y a veces con otro significado. La configuración del servidor Samba es responsable de mantener la coherencia entre sus dos "personalidades", aunque pueden haber casos concretos, donde el administro de la red conscientemente utiliza la dualidad, para darle por ejemplo permisos de acceso diferidos a clientes provenientes de diferentes subnodos.

Nótese, que el transporte TCP/IP para NetBios también permite crear redes Microsoft que exceden los límites de una red local.

**Permisos y usuarios** Samba mantiene una pase de datos de usuarios y sus claves de acceso aparte de los mecanismos de autentificación Unix. Hay previsión para que la base de datos de Samba y la de Unix se sincronizen, lo que permite a las y los usuarias/os cambiar la clave de acceso de la red inclusive desde computadoras MS-Windows. Esto se utilizará cuando la computadora que alberga el servidor Samba también provee servicios para estos usuarios. Otra situación opuesta puede ser, que una computadora Linux se usa como controlador de dominio, donde se albergan las "cuentas SMB" de manera virtual y probablemente se tiene registrado muchas/os usuarias/os de la red Microsoft, de los cuales ninguna/o tiene una cuenta Unix en esta computadora.

Los servicios que son declarados para compartirse en red en primer lugar se manejan mediante los mecanismos de control de acceso del protocolo SMB, pero a continuación también aplican los permisos de acceso de Unix. Esto significa que aunque un directorio puede estar compartido en la red aparentemente en modo escritura/lectura, pero al momento de conectarse desde una computadora remota el núcleo solo permite acceso de lectura a los recursos indicados.

Samba puede actuar como controlador de dominio, pero también puede obtener la información sobre la autentizidad de una sesión desde un controlador remoto.

**Notación de nombres** Según la versión de protocolo y cliente SMB cambia el concepto del trato de Nombres de usuarios, claves y servicios compartidos. Existe la posibilidad que cualquiera de estos antes de transmitirlo al servidor sea convertido en mayuscula, e inclusive la longitud permitida varia, y nombres o claves largos pueden ser truncado o convertido parcialmente. En la configuración hay posibilidad como circunvar algunos de los problemas que resultan, pero también es recomendable, según los versiones de los clienes concretos observar reglas específicas para la selección de nombres de servicios, usuarios y claves.

En las primeras versiones de redes Microsoft las claves de acceso fueron guardados y transmitidos en texto claro, lo que no es una idea muy buena. A continuación fueron introducidos mecanismo de encripción, que también son diferidos. Especialmente Windows NT ha tenido que implementar esto como medida de seguridad, pero obviamente tiene que mantenerse la opción de inviar o recbir claves no encriptados de parte de clientes "antiguos".

En un sistema Unix no se considera deseable que un/a usuario/a puede abrir el acceso a recursos hacia la red, y Samba comparte este concepto. Al contraste con los sistemas

operativos Microsoft decendiente de Windows 3.11 un/a usuario/a no puede compartir recursos, sino todos los SHARES tienen que ser declarados en el archivo `smb.conf`, contenido comunmente en el directorio `/etc` o `/etc/samba`. Este archivo tiene una estructura parecido a los archivos de configuración "`.ini`" de MS-Windows:

```
[nombre]
parametro = valor    ; comentario
```

Está dividido en secciones nombrados, los límites de la sección se determinan con el nombre encierrado en parentesis cuadrados. Los parámetros so nombran y se les asigna un "valor" mediante el símbolo "=". Parámetros no mencionados no se definen u obtienen valores por defecto, así que cada sección puede tener una cantidad variable de líneas de parámetros.

Hay dos tipos de secciones: [global], [homes] y [printers] son nombres reservados, todas las demas secciones son *nombres* de SHARES, de servicios o recursos compartidos.

La sección [global] es obligatoria y define parámetros generales, como por ejemplo el nombre NetBios, o la versión del protocolo.

### SHARES automáticos

Las impresoras definidas en printcap y los directorios home de las y los usuarias/os son elementos constantes en un sistema Unix. Los shares automáticos proveen a usuarios en la red que tienen una cuenta unix en la computadora local acceso a estos recursos, que además tuvieran en una sessión logín Unix automáticamente, desde clientes Microsoft de una manera natural y fácil de configurar.

La presencia de una sección [home] activa la creación automática de uns SHARE en el momento que un/a usuario se autentifica mediante SMB en el servidor Samba local y cuando este usuario tenga una cuenta unix, y por lo tanto un directorio home. La sección [home] sirve entonces como patrón para los parámetros de esta compartido. Se puede permitir acceso de solo lectura, se puede hacer el directorio invisible (no "browsable"), se puede determinar el modo de acceso en cual se crean archivos si se accede a este directorio, etc. Para un usuario que accede a la computadora a través del "Ambiente de Redes" se presenta la computadora y un compartido "homes" dentro del cual va a encontrar los archivos y subdirectorios de su propia cuenta Unix en aquella computadora. Esta apariencia es igual para todos y todas las usuarios/as en la red, aunque efectivamente acceden a diferentes subdirectorios en la computadora Linux.

Este mecanismo puede usarse, para convertir una computadora Unix como servidor de archivos privados en una red Microsoft con un esfuerzo de configuración mínimo.

De una manera parecida y dada la presencia de una sección [printers] Samba analiza todas las entradas en el archivo printcap y los convierte en SHARES - colas de impresión Microsoft, con los parámetros indicados en esta sección.

### SHARES explícitos

Supongamos, que queremos proveer un directorio en red, al cual pueden acceder todas las computadoras Windows para intercambiar archivos de una computadora a otra y que utilizamos /usr/share/public como este directorio, tenemos que dar permiso de acceso de escritura y lectura al proceso smbd, y crear una sección en `smb.conf`, por ejemplo:

```
[public]
path = /usr/share/public
writeable = yes
browsable = yes
```

Aunqe esto es probablemente la forma más sencilla de crear un SHARE también es una forma sencilla de obtener problemas por abuso consciente o inconsciente.

Si queremos restringir el acceso a un grupo definido de cuentas podemos usar el parametro:

```
valid users = jefe, secre, venta
```

Que en este caso solamente permite a los usuarios "jefe", "secre" y "venta" acceder al directorio público. Esto implica, que estos usuarios existen en la computadora local, o como usuarios Samba. Suponiendo que estamos usando este archivo en la computadora "jefe.cigarros.com" podemos ganar acceso de otra computadora mediante el nombre "`\\jefe\public`", mientras no hayamos cambiado el nombre NetBios de la computadora.

Los parámetros por sección son muchos, vamos a presentar solamente algunos comunes:

**path** designa la vía de acceso, o en caso de una impresora el directorio de cola a usar

**writable** Accede permiso de escritura a usuarias/os conectadas/os

**readable** es un antónimo para "writeable". Utilizar "writable = no", y "readonly = yes" es equivalente. Muchos parámetros tienen antónimos o sinónimos, lo que puede usarse para aumentar la comprensión del archivo de configuración (o para hacerlo más confuso)

**readable** Accede permiso de lectura

**browsable** permite incluir el servicio en el Browse List. Si browsable es "no", el servicio solo puede accederse, sabiendo su nombre completo.

**printable** indica que el servicio es una impresora

**allow hosts** permite especificar, cuales computadoras remotas tienen acceso al recurso

**deny hosts** rechaza la conexión desde las computadoras especificadas

**comment** define el texto adicional mostrado para clientes en el Browse List.

**create mode** especifica una máscara octal de permisos, con la cual son creados los archivos, p.ej. una máscara de 0600 especifica que solo el/la usuario/a tendrán acceso de escritura y lectura a archivos que crean en un directorio compartido.

**guest ok** permite acceso sin especificar una clave

**guest account** especifica el usuario unix a utilizar para accesos desde una conexión anónima.

**guest only** especifica, que solo se puede acceder como el usuario guest al servicio.

No es necesario crear un archivo de configuración para samba manualmente. Hay al menos dos aplicaciones que permiten crear y mantener al día al archivo smb.conf. Gnomba es una aplicación del Gnome Desktop, que permite editar y crear secciones individuales de smb.conf, y proporciona en cada sección los parámetros apropiados y sus respectivos valores permitidos.

#### Configuración de Samba mediante Swat

Samba Web Administration Tool o "Swat" es la utilidad de configuración nativa para Samba. Swat opera como un servidor Web, que se conecta por defecto en el puerto TCP/IP 901. Al navegar la dirección:

```
http://localhost:901
```

El/la usuario es presentado/a con un diálogo que pido nombre y clave. Si se autentifica como root gana acceso a toda la configuración de Samba, si se autentifica como cualquier otro usuario el acceso está restringido a la revisión de parámetros y obtener información sobre el estado del servidor. Si la documentación de Samba está instalada en formato html en la computadora, también ésta puede ser navegada, proveyendo una ayuda sensitiva al contexto de la operación a efectuar, mediante hyperenlaces.

Swat presenta una página "home" con una "barra" de tareas que incluye la administracón de: la sección global de smb.conf, los , SHARES, y de las usuarios y sus claves. Además puede revisarse el estátus del server y las conexiones actuales de clientes y la documentación, y se pueden parar y reanudar los daemon-ios smbd y nmbd.

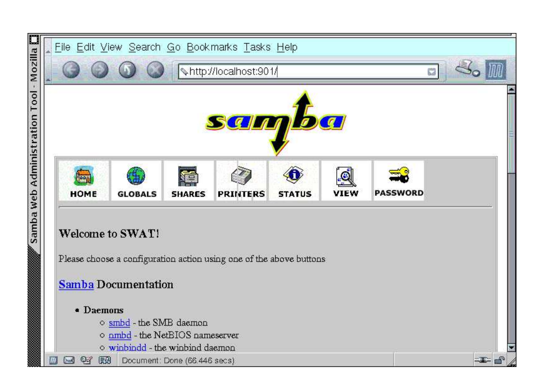
La imágen 10.1 muestra una parte de la portada de Swat.

Una de las ventajas de Swat es, que la Samba puede principalmente ser configurado desde cualquier computadora en la red, sin embargo tienen que tomarse en consideración que la clave de autentificación queda expuesta en texto claro al menos que se aplica alguna medida de seguridad adicional, y generalmente es prudente solo permitir la configuración Swat desde la computadora local.

## 10.2. Redes Novell - ncp

El protocolo desarrollado por Novell para la implementación de sus redes es IPX, y basandose en este SPX. Para compartir archivos se utiliza IPX y NCP - el "Network

Figura 10.1.: Página principal del cofigurador Samba - Swat




Core Protocol", para utilizar impresoras remotas es suficiente el protocolo ipx. El núcleo de Linux provee módulos para ambos protocolos, ipx y ncpfs. Para hacer funcionar la computadora Linux como servidor para clientes Novell es necesario una configuració específica del núcleo, pero además se necesitan los programas respectivos para acceder a la funcionalidad. Existe el paquete Linware y el paquete mars_nwe, que implementan las funciones necesarias para poder presentar las impresoras y/o directorios de una computadora Linux en la red como se procedieran desde un servidor Novell. NetWare provee en muchos casos las restricciones de nombres de MS-DOS, con nombres de archivos solamente mayuscula y con ocho letras para el nombre y tres letras para la extensión. También para NetWare se puede conseguir programas que corren en MS-Dos de manera gratis, lo que permite conectar computadoras viejas de clase 286 o inferior con MS-DOS como sistema operativo, y usarlos como terminales, o como servidores de impresoras.

IPX es un protocolo de Red más elaborado que SMB y los sistemas operativos Windows implementan programas Servidores y Clientes para el, lo que puede ser usado como alternativa a la conexión mediante Samba.

Una diferencia entre IPX y Samba es, que IPX no utiliza un protocolo de transporte, sino provee su propio gestor para el enrutamiento y transporte de paquetes de información.

Al momento de considerar usar una computadora Linux como "puente" entre redes Novell y redes Microsoft debe planificarse cuales recursos se ofrecen cada red. Comunmente se querra hacer los mismos directorios y las mismas impresoras de la computadora Linux disponible en ambos redes, y darles además nombres iguales o parecidos para facilitar a las y los usuarios de la red su operación.

## 10.3. Redes Appletalk netatalk

También para la conexión con computadoras MacIntosh en ambas direcciones hay previsiones en el núcleo de Linux, que implementan el protocolo AppleTalk y los gestores para los sistemas de archivos de este sistema operativo. También como en las otras redes

se requiere la instalación de un paquete de programas externos, en este caso llamado "NetaTalk".

AppleTalk contiene un abanico de protocolos de red, que son manejados por netatalk, que provee aplicaciones parecidos a mars_nwe y Samba para que Computadoras MacIntosh puedan usar impresora y archivos de Linux, o para que deste Linux se puede imprimir o grabar archivos en impresoras o discos de las computadoras MacIntosh.

# 11. Usuarios

## 11.1. Cuentas especiales, root, sysadmin

## 11.2. Cuentas normales

## 11.3. Grupos de trabajo

## 11.4. Authorización en la red

[opcional]

## 11.5. Como no ser root

[Recetas para adminstrar un sistema Linux en la vida diaria. Instalación de herramientas sudo, calif, etc., importante pero opcional.]

# 12. Linuxconf

[herramienta homogénea de configuración]

# Parte III.

# Instalación

# 13. Proceso de arranque

[Qué elementos tiene un sistema operativo corriendo,
Cómo se llega a este estado ->mapa para las herramientas de instalación del sistema ]

# 14. Disquetes de arranque y rescate

[Comparación Debian, Redhat, syslinux, lilo y autoarranque del núcleo]

# 15. Disco duro

[Preparación del disco duro, planificación de particiones]

## 15.1. Partitiones y sistemas de archivos

[fdisk, y formateo, swap-partitions]

## 15.2. Lilo y grub

[grub solo opcionalmente. Lilo, instalación, configuración]

# 16. Instalación del Software

[Instalación del sistema base. Qué son paquetes precompilados de Software]

## 16.1. Redhat

[opcional]

## 16.2. Debian

[Instalación CD y Red]

## 16.3. Perfiles, Paquetes, Fuentes

[Perfiles de computadoras según su uso. Selección predeterminada de paquetes. Instalación via mirrorring.

Dependencia de paquetes, problemas con incompatibilidad y versiones, Instalación desde el código fuente. ]

# 17. Hardware específico

En este capítulo se explica, como se puede activar dispositivos que no están incluidos en la configuración básica de las distribuciones. También se indica la forma como optimizar y reinstalar el núcleo del sistema operativo mismo.

## 17.1. Núcleo modular

Una fundamental característica del núcleo de Linux es su modularidad. Mediante el mecanismo de módulos del núcleo se pueden conseguir dos efectos:

- Se reduce el tamaño de la imagen del núcleo (el archivo que se carga al arrancar el sistema operativo)
- Se pueden re-inicializar driver para ciertos dispositivos de hardware, o usar consecutivamente diferentes drivers para el mismo dispositivo sin recargar todo el núcleo (sin reiniciar la computadora).

### 17.1.1. Objetos dinámicos

En la programación se utilizan bibliotecas de programas para resolver tareas comunes. P.ej. una de las más famosas biblioteca en la programación en C es asociada con el archivo de inclusión <stdio.h>. Es la biblioteca que contiene las rutinas para imprimir y leer caracteres desde archivos y dispositvos.

A la hora de escribir un programa, el o la programador/a ya no tienen que reinventar las rutinas de entrada y salida, sino utilizan las prefabricadas de la biblioteca.

Como casi todos los programas en el sistema incluyen esta biblioteca de rutinas, a la hora de cargarlos se desperdicia espacio de memoria, porque el código de programa de las rutinas de la biblioteca está incluido en cada programa. Para evitar esta desventaja se inventaron las bibliotecas dinámicas, u objetos compartidos. El primer nombre (DLL: dinamic linked library = biblioteca enlacada dinámicamente) proviene de los sistemas operativos Microsoft, mientras en Linux se utiliza la segunda expresión: Shared Objects = objetos compartidos, designados con la extensión de archivo ".so". Las bibliotecas absolutas, o estáticas, que se requieren solamente para la compilación tienen la extensión ".a".

En la compilación de programas estáticos se incluyen los archivo de las bibliotecas absolutas después de generar el código del programa mismo, y a continuación se resuelven los saltos a las rutinas de la biblioteca mediante un programa llamado el Linker = enlazador. El resultado es un archivo monolítico que contiene todo el código necesario para ejecutar su tarea.

En contraste se compilan programas modulares sin incluir el código de las bibliotecas. El Linker ahora tiene un rol diferente, y en efecto es un programa diferente, en Linux: “ld.so”, shared object linker. A la hora de cargar un programa con bibliotecas dinámicas ld.so entra en acción, analiza cuáles bibliotecas requiere el programa para funcionar, los carga en espacios libres en la memoria y en este momento modifica el código del programa de tal manera, que todos los enlazes resuelven correctamente. Si la biblioteca ya está cargado en la memoria, ld.so no vuelve a hacerlo, sino resuelve los saltos hacia las rutinas correspondientes hacia la biblioteca presente. Esto requiere, sin embargo, que las bibliotecas estén escritas de una forma que permite reutilizar las mismas rutinas por diferentes programas.

Ahora comparemos módulos estáticos con módulos dinámicos:

Ventajas de módulos dinámicos:

- requieren menos memoria, total, si mas programas las comparten
- si una biblioteca tiene una falla, se puede corregir sin necesidad de recompilar todos los programas. Solo se sustituye la biblioteca respectiva.

Desventajas de módulos dinámicos:

- Las veces que se carga un programa, se tiene que resolver de nuevo las direcciones: pérdida de tiempo.
- Si se copia un programa a otra computadora, también tienen que copiarse todos los archivos de bibliotecas de los cuales depende.
- cada programa y cada biblioteca crece un poco de tamaño, porque tiene que contener información adicional para su enlace dinámico exitóso.
- Se requiere el programa ld.so (y su adecuada configuración) para poder ejecutar un programa.

En el sistema Linux es necesario ejecutar la utilidad “ldconfig” como root, cuando se instalan nuevas bibliotecas. El archivo “/etc/ld.so.conf” contiene una lista de directorios que se utilizan para encontrar y registra bibliotecas dinámicas. Esta lista solamente es para directorios adicionales, ya que ld.so “sabe” las vías de acceso estándard.

### 17.1.2. Módulos del núcleo

El núcleo del sistema operativo es por un lado también un programa C, pero por otro lado es especial, en el sentido que se carga desde “la nada”, y no puede contar con muchas de las utilidades versátiles, que el sistema operativo pone a disposición. Sin embargo se integró al núcleo de Linux un sistema de resolución dinámica de “bibliotecas”, que en este caso se llaman módulos. Esto quiere decir, que ciertas rutinas del núcleo no se incluyen en el mismo archivo que el bootstraploader - el cargardor inicial lee del disco duro (o de donde sea), sino que residan en el disco duro en archivos aparte. A la hora que se requiere

estas rutinas se tienen que copiar estos módulos del disco duro a la memoria principal, y resolver las direcciones dentro del núcleo a la ubicación que tengan en este momento. En la figura 17.1.2 se ilustra la diferencia entre un núcleo estático y un núcleo con módulos.

Usar un núcleo con módulos implica, que primero tiene que funcionar el acceso al disco duro (o más preciso: al sistema de archivos) para que se pueden cargar módulos adicionales y por lo tanto que las rutinas esenciales para acceder al sistema de archivo no pueden ser compilados como módulos. Además el núcleo necesita "saber" cuáles módulos pueden potencialmente cargarse "en el". A la hora de compilar el código fuente del núcleo de Linux se le indica, cuales rutinas son residentes (fijos) en el, cuáles se compilan como archivos de módulos y cuales no se compilan. Para los que son módulos se reservan los saltos para cargarlos en el núcleo, por lo que se agrandece mínimamente el volumen del núcleo.

Ventajas de módulos de núcleo:

- Se puede reducir el volúmen del archivo de núcleo:
  - así es posible grabar un archivo núcleo en un disquete, que tiene menos capacidad de almacenamiento que el núcleo total.
  - el núcleo reducido cuenta con un tiempo de carga reducido, aunque este tiempo se invierte (y con interéses) a la hora de cargar el núcleo.
- Se puede mejorar el rendimiento de la memoria virtual, descargando rútinas del núcleo de la memoria, que no se requieren en un determinado momento.
- Se puede descargar y volver a cargar módulos, y de esta manera re-inicializar dispositivos
- Se puede usar un dispositivo de diferentes maneras. Ej.: el puerto paralelo puede usarse para plip, impresión, discos externos (zip-drives)[1]. Para esto se descarga el módulo en uso, y después se carga el otro.

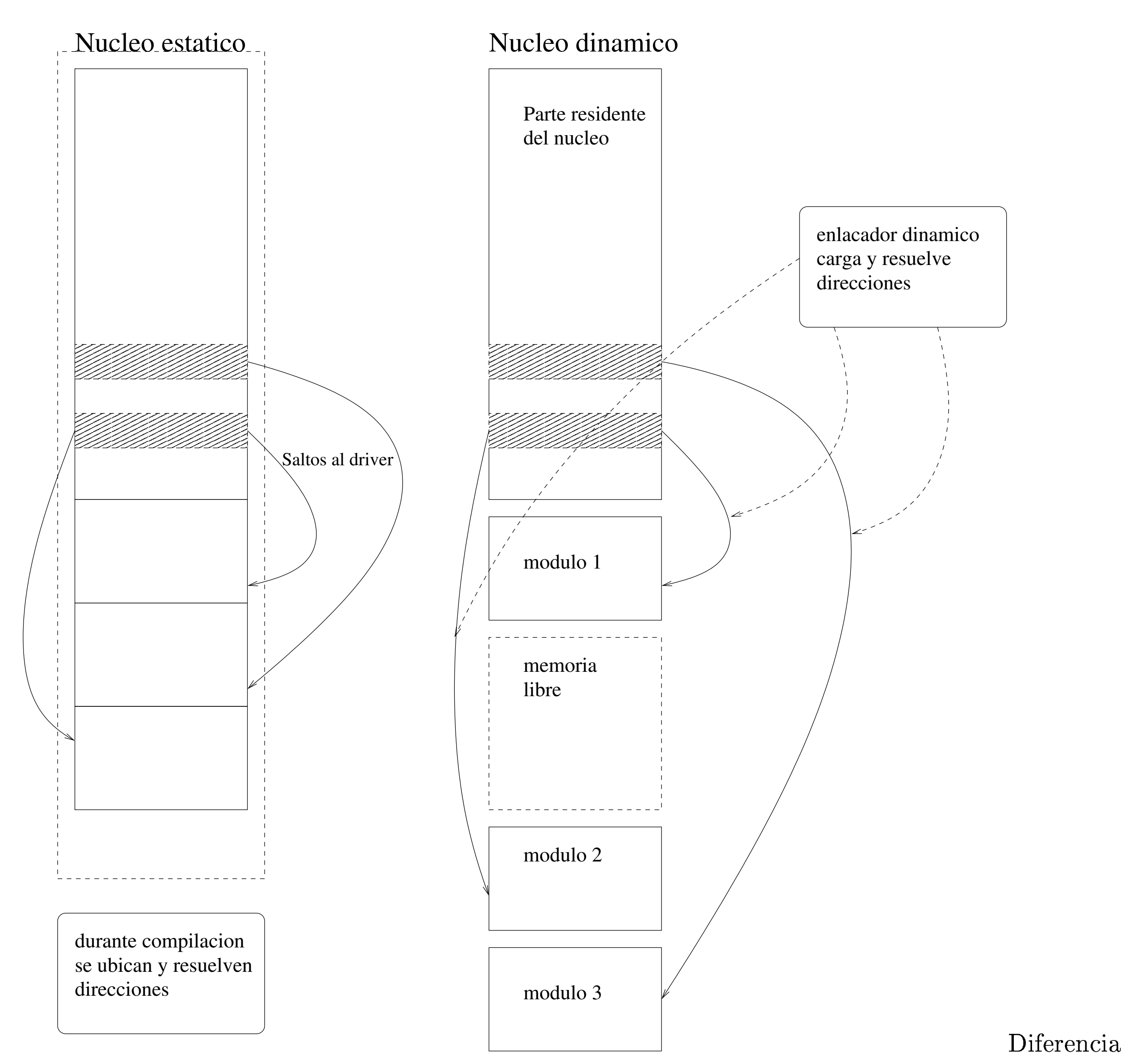
Los mismos módulos pueden ser "modulares" por ellos mismos, quiere decir, si varios módulos utilizan las mismas rutinas, estos pueden separarse en un módulo aparte, y se dice que entonces estos módulos dependen del último. Tiene que cargarse primero el módulo con las rutinas compartidas, para poder cargar los otros.

El núcleo puede ser configurado también para cargar automáticamente los módulos que necesita. Esta funcionalidad se llama kernel-daemon o kerneld.

Hay varias utilidades para cargar, descargar y administrar los módulos :

**insmod** ***via de acceso*** carga un archivo módulo de núcleo en la memoria. La extensión de estos archivos es ".o". En la líne de comando se especifican los parámetros que son específicos para este módulo.

[1] En el caso del puerto paralelo está resuelto el uso múltiple de una manera mas elegante en los núcleos de versión 2.2 y mayores. Hay un módulo para el puerto, el cual es usados por otros módulos. Si un módulo accede al módulo del puerto bloquea los otros, cuando termina de usar el módulo lo cede a los demás módulos. No se necesita descargar y recargar los módulos individuales.




Diferencia entre núcleo estático y núcleo modular.

**lsmod** alista todos los módulos actualmente cargados en la memoria.

**rmmod *nombre de módulo*** remueve un módulo del núcleo. El nombre del módulo es comunmente el nombre del archivo, pero sin la extensión ".o".

**modinfo *via de acceso*** muestra información sobre el módulo indicado por <via de acceso>. En especial se puede averiguar con este comando, cuales parámetros son acceptados o requeridos por el módulo. También se puede utilizar el nombre del módulo en vez de la vía de acceso.

Y existe todo un sistema de gestión de módulos, llamado modutils, que facilita poner en orden las dependencias de los módulos. En un sistema instalado, estas utilidades por lo general no se necesitan invocar manualmente, porque están integrados en el arrance y la administración automático.

**depmod** crea un archivo de dependencias para todos los módulos en el disco duro.

**modprobe *nombre de módulo*** modprobe utiliza esta lista para cargar el módulo indicado por *nombre de módulo*, pero carga anteriormente todos los módulos del cual depende el indicado.

En el archivo `/etc/modules.conf` pueden especificarse además varios parámetros que se aplican a la hora de cargar un módulo. Si se trata de módulos que acceden directamente a un dispositivo de hardware, muchas veces se requiere especificar el puerto de Entrada/Salida o la interrupción que utiliza el dispositivo. Especialmente si pueden haber varios dispositivos del mismo tipo en una computadora, se puede especificar el orden en el cual se les asigna su numeración. Otro comando utilizado con frecuencia en este archivo es `alias` que permite darle un "nombre" sinónimo a un módulo. Así se puede por ejemplo llamar al módulo del hardware de sonido siempre "sounddriver", independientemente si es un soundblaster, o si es un chipset cmpci.

El comando "insmod" requiere, que se especifíca las opciones en la línea de comandos, mientras modprobe los lee desde el archivo modules.conf. Con el tiempo la cantidad de módulos ha crecido a un gran número, así que el sistema Debian Gnu/Linux utiliza un mecanismo de diversificación de estos archivos. En el directorio /etc/modutils se pueden crear archivos o editar los existentes, en los cuales se especifican opiones o comandos correspondientes a modules.conf. Con el comando "update-modules" todos estos archivos son integrados en el archivo modules.conf automáticamente, el cuál no debe ser editado a mano.

En el sistema Debian Gnu/Linux se utiliza el archivo /etc/modules para indicar, cuales módulos deben cargarse automáticamente al iniciar el sistema. Se escribe una línea por cada módulo, solamente con el nombre del módulo que se quiere cargar.

Ya que un sistema puede trabajar con diferentes versiones de núcleos existe un esquema que permite a los modutils identificar cual módulo cargar. Los módulos están ubicados en el directorio /lib/modules, en un subdirectorio con el nombre de la versión del núcleo, para el cual fueron compilados. Si tenemos un núcleo de la versión 2.2.17 (Debian 2.2)

y uno de la versión 2.4.0 entonces encontramos dos directorios: /lib/modules/2.2.17 y /lib/modules/2.4.0[2].

Bajo estos directorios hay una jerarquía de subdirectorios, para dividir los archivos de módulos según su función o tarea. Para detectar cual versión de núcleo estamos usando puede ejecutarse la línea de comando: uname -a, o uname -r

## 17.2. Recompilación del núcleo

Pueden haber varias razones por qué recompilar el núcleo del sistema operativo, que parten todos del hecho que el núcleo "universal" o genérico que se instala en el proceso de la instalación del sistema no es adecuado para la tarea que se quiere realizar con la computadora. Pueden haber uno varios de los siguientes escenarios:

- La computadora utiliza dispositivos de hardware para los cuales no existían los gestores en el tiempo cuando fue fabricado el núcleo genérico.
- La computadora utiliza dispositivos de hardware pocas veces usados y que no son habilitados en el núcleo genérico.
- Se quiere compilar estáticamente un gestor que está actualmente compilado como módulo o vice versa.
- La aplicación de la computadora requiere facilidades o parámetros especiales, que no son habilidados en el núcleo genérico.
- Se quiere instalar un núcleo más reciente para utilizar funcionalidades no incorporadas en el núcleo genérico.
- Se quiere reducir el tamaño del núcleo y/o la cantidad de módulos, solamente incorporando los gestores necesariamente requeridos para el sistema/la aplicación. Esto puede ser útil para liberar memoria para las aplicaciones, o puede ser necesario si la computadora dispone de muy poca memoria principal.

El procedimiento a realizar ha llegado a ser bastante estándardizado y no requiere conocimientos de programación. El primer paso es la instalación del código fuente del núcleo, por lo general (y recomendablemente) en un subdirectorio de "/usr/src". En el sistema Debian el núcleo se instala en un subdirectorio llamado "kernel-source-<version>", en RedHat se utiliza "linux-<versión>" y se crea un enlace simbólico "linux" al subdirectorio del núcleo actualmentente utilizado.

Para trabajar con el núcleo hay que entrar en este directorio, que se llama también el directori raíz de la fuente del núcleo. El siguiente paso es la configuración de todos los parametros y la definición de los gestores estáticos y/o módulos de núcleo que se quiere incluir. Para esto hay tres utilidades:

[2] Si compilamos dos versiones diferentes de un núcleo de la misma serie, tenemos que distiguirlos y también hay que usar directorios diferentes para los núcleos. Eso es posible con un anexo al número de versión, que se especifica cuando se compila el núcleo.

1. Modo texto solamente: "make config"
2. Modo diálogo, con un interfáz gráfico en la consola de texto: "make menuconfig"
3. Modo gráfico, con un interfáz Tcl/Tk en X-windows: "make xconfig"

La primera versión no es recomendable, ya que procesa todas las preguntas de configuración de forma secuencial y no permite regresar a decisiones anteriores. La segunda y la tercera versión permiten seleccionar las categorias interactivamente y repetir los pasos de configuración cuantas veces sea necesario. Además permiten una cierta administración de configuraciones, ya que permiten cargar y salvar archivos de configuración nombrados. Es perfectamente posible y a veces inclusive recomendable compilar en una computadora núcleos para otras computadoras. El código fuente actualmente alcanza los 80 Megabyte de espacio y se requier alrededor de un cuarto a una media hora con una computadora veloz con bastante memoria para compilarlo. Así que es preferible compilar núcleos para computadoras lentas en otras computadoras y después copiarlos a la máquina destino. Para esto se maneja p. ej. archivos de configuración con los nombres de las computadoras en la computadora donde se compilan los núcleos, para más rápido acceso a las configuraciones diferentes.

Al finalizar la configuración se graba un archivo ".config" que contiene todos los parametros de compilación necesarios. Ahora se procede a compilar el núcleo, esencialmente con los comandos:

```
make deb; make clean; make vmlinuz; make modules; make modules_install
```

Se utiliza preferiblemente una sola línea de comando, ya que cada uno de los procesos es tardado y de esta manera no se tiene que estar presente para dar los comandos individuales a concluir cada paso. Sin embargo, si la ejecución de un paso se interrumpe por un error, un hecho que pocas veces ocurre, se tiene que reinicializar solamente a partir del comando fallado.

La compilación del núcleo es una prueba dura para el procesador y para la memoria principal de la computadora. Es casí imposible que una memoria con fallas no se descubra a la hora de (re)compilar el núcleo. El símptoma presentado en este caso es muchas veces un mensaje raro con "broken pipe" o "compiler error", o puede ocurrir en la fase de generación del código de máquina, que normalmente nunca falla. Con otras palabras: Si duda de la memoria de su computadora compila un núcleo de Linux en ella. Otra razón para la compilación que no mencioné arriba.

Para usuarios de un sistema Debian Gnu/Linux las cosas se facilitan con el paquete utilitario "kernel-package", que provee el comando "make-kpkg". Para compilar un núcleo y preconfigurarlo para la instalación solamente se necesita ejecutar:

```
make-kpkg kernel-image
```

y eventualmente de antemano "make-kpkg clean".

### 17.2.1. Parametros de configuración

Hay varios categoría de configuraciones para un núcleo, las cuales cada una tiene su cantidad de opciones. Muchas de las categorías o son presentes, o no lo son. En caso que se selecciona por ejemplo compilar un núcleo sin soporte de red, ya no se hacen preguntas acerca de tarjetas de red, protocolos de red, o sistemas de archivos que permiten compartir archivos en red.

A continuación se describe brevemente cada categoría de configuración del núcleo versión 2.2.17 que es estándard para la distribución Debian Gnu/Linux. Esto está tomado de la configuración mediante "make menuconfig". En el programa de configuración mismo se puede obtener ayuda extensiva para cada categoría y para cada pregunta (opción), pero también hay mucha infomración en el código fuente del núcleo mismo. En primer lugar esta se encuentra en el directorio "Documentation" dentro de la raíz del núcleo, después hay información adicional en los directorios de los archivos .c mismos.

**Code maturity level options** permite activar o desactivar opciones en todas las otras secciones, que no se consideran todavía estábles. Para considerar un driver, o una opción de configuración como estable tiene que pasar un tiempo de prueba considerable, así que muchas veces vale la pena habilitar esta opción, ya que provee gestores considerablemente estables.

**Procesor type and features** en esta sección se selecciona el tipo de procesador para el cual se compila el núcleo. Si se quiere compilar un núcleo genérico posiblemente se considera seleccionar un procesador 486 o máximamente Pentium. Seleccionando un procesador más avanzado optimiza el código creado, pero el núcleo resultante no puede correr en un procesador inferior.

**Loadable module support** aquí se habilita el uso de módulos de núcleo. En principio se pueden utilizar los mismos archivos módulos para diferentes versiones de núcleo, aunque eso no siempre funciona. En esta sección se define si solamente se permite cargar módulos con la versión precisa del núcleo, que requiere un chequeo adicional (y una pequeña pérdida de espacio y tiempo) o si se carga los módulos sin chequeo.

**General setup** en esta sección se activa y desactiva diferentes subsistemas, como redes, soporte para diferentes buses de periféricos (PCI, MCA), etc.

**Plug and Play support** habilita soporte para la autodetección y/o autoconfiguración de puerto de Entrada/Salida de los dispositivos de hardware en buses ISA y PCI. Nota: Plug and Play o PnP (Enchufa y juega) del bus ISA a veces también se traduce con Plug and Praise (Enchufa y comienza a rezar).

**Block devices** permite seleccionar los gestores para floppy, disco duro y otros dispositivos de almacenamiento de datos. También se puede habilitar aquí el uso de múltiples discos en configuraciones RAID, que provee aumento de capacidad, acceleración del acceso, o seguridad mejorada en los discos duros.

**Networking options** configura el soporte para diferentes facilitades y protocolos de red, y opciones para adaptar los gestores a computadoras muy rápidas o muy lentas. En especial, aquí se activa el soporte para IP-Masquerading (NAT) y para Firewall, opciones que permiten configurar la computadora como servidore especializados en conectar una red local al Internet.

**Telphony Support** para activar el gestor para dispositivos de telefonía.

**SCSI support** para activar el soporte para diferentes tipos de dispositivos SCSI (discos, cintas, CDROM, otros), y para seleccionar los gestores para la tarjeta SCSI que se utliza. Si se quiere usar ZIP-Drives de Iomega que utilizan el puerto paralelo se tiene que utilizar esta sección, ya que estas unidades utilizan una simulación del puerto SCSI via el acceso al puerto paralelo.

**I2O device support** I2O es una tecnología de dispositivos genericos de entrada/salida que relevan la CPU del trabajo de acceder directo a los controladores físicos y aumentan la velocidad total del sistema. Esta opción solamente se requiere si la computadora utiliza este tipo especial de hardware.

**Network device support** para seleccionar los gestores para los dispositivos de red en la computadora.

**Amateur Radio support** esta sección ofrece la selección de los diferentes protocolos de red que se utilizan para el intercambio de datos via packet radio, y los gestores para los dispositivos disponibles de esta forma de conexión.

**IrDA (infrafred) support** gestores para los enlaces de infrarojo.

**ISDN subsystem** Integrated Services Data Network es una norma ampliamente usado en Europa que facilita varios canales de intercambio de datos y dos canales de audio en paralelo mediante la línea telefónica.

**Old CD-ROM drivers (not SCSI, not IDE)** hay algunas unidades lectoras de CD-ROM que utilzan tarjetas especiales para su control, en esta sección se encuentran los gestores para ellos. Una sección que prácticamente nunca se utiliza.

**Character devices** Los gestores para todos los dispositivos de carácter. En especial para la console (monitor y teclado), los puertos seriales y los puertos de teletype virtuales "pty" que no corresponden a un dispositivo de hardware, estos se usan extensamente para el sistema X-Windows. También se encuentran aquí los dispositivos para captura de televisión.

**Filesystems** en esta sección se seleccionan, cuales sistemas de archivos serán disponibles para el núcleo. Se pueden seleccionar diferentes sistemas de archivos para medios físicos, y sistemas de archivos en red, o sea, gestores que permiten acceder a archivos remotos de una manera transparente, como si estos estuvieron montados en un directorio local.

**Console drivers** prove opciones para las tarjetas de video y para activar el soporte para Frame buffers devices. Estos últimos son dispositivos virtuales, que proveen una presentación equivalente de la pantalla independientemente de la arquitectura (el procesador) del sistema.

**Sound** permite seleccionar la tarjeta de sonido que se está usando.

**Kernel hacking** activa una opción para programadores de núcleo, que permite ver los registros del procesador mediante una combinación especial de teclas.

**Load an Alternate Configuration File** carga un archivo de configuración nombrado

**Save Configuration to an Alternate File** salva la configuración actual en un archivo nombrado.

En la versión 2.4. del núcleo la estructura es un poco diferente, aquí solamente se discutan opciones no existentes en el núcleo 2.2.

**Memory Technology Devices (MTD)** soporte para memoria Flash, etc. esto es memoria no volátil de grán volúmen que puede usarse como "discos estáticos".

**Parallel port support** para el driver genérico, en dependencia de la arquitectura de computadora, el cual va usado para el gestor de impresión, discos paralelos, Plip, etc.

**Multi-device support (RAID and LVM)** vea la opción "Block devices" arriba.

**ATA/IDE/MFM/RLL support** El soporte para diferentes gestores de nivel bajo de discos fue separado en esta sección, en 2.2. esta en "Block devices".

**Fusion MPT device support** para computadoras que usan este dispositivo. Es una tecnología que conecta los dispositivos de entrada/salida mediante fibra óptica a la CPU.

**Input core support** habilita el soporte para dispositivos USB de interacción human - teclados, joystick, etc.

**Multimedia devices** para dispositivo de procesamiento de señales de video; en 2.2.17 estas opciones están en "Character Devices".

**USB support** gestores para los dispositivos hardware de USB = Universial Serial Bus, una tecnología que permite concatenar teclados, ratones, impresoras, escaner, discos externos etc. en un solo cable serial.

**Bluetooth support** son gestores para esta clase de dispositivos USB.

### 17.2.2. Versiones de núcleos

## 17.3. Instalación de un núcleo

En un núcleo modular hay varios componentes que poner en su lugar para que el sistema pueda utilizarlo en el arranque. Primero lógicamente el archivo de núcleo mismo, y un bootstrap loader que lo carge a la memoria en el inicio del sistema. Este va a ser lilo en nuestro caso. Después se necesitan copiar los archivos de los módulos correspondientes al núcleo al directorio respectivo. y por fín conviene copiar el archivo con los símbolos de rutinas exportádos del núcleo a su lugar correspondiente, porque varias utilidades administrativas los utilizan para mostrar información sobre el sistema.

En la misma computadora se puede utilizar varios núcleos, inclusive utilizando el mismo dispositivo para el sistema de archivos raíz.

### 17.3.1. Lilo

Aunque en teoría se puede utilizar cualquier lugar en el sistema de archivos que esté disponible al Bios de la computadora para guardar el archivo del núcleo, existen convenciones que preferiblemente se respetan. Hay una convención de tener un enlace simbólico con nombre "vmlinuz" en el directorio raíz. El directorio "/boot" se utiliza para guardar todos los núcleos que se ofrecen para el arranque. El núcleo estándard se instala entonces con un archivo /etc/lilo.conf estándard que contenga las líneas:

```
image=/vmlinuz
      label=Linux
      read-only
```

y mediante el comando: "lilo"

Una convención práctica del sistema Debian Gnu/Linux es, tener un segundo enlace simbólico /vmlinuz.old que indica al núcleo genérico de instalación en el directorio /boot y que se puede cargar con la etiqueta "LinuxOld". De esta manera se garantiza que siempre hay un núcleo "de reserva" si se compila e instala un núcleo nuevo que no funciona.

### 17.3.2. Módulos

Con el comando "make modules_install" ya se copian los archivos de módulos generados a la hora de compilar el nucleo en su lugar correspondiente. No es trivial ubicar dónde específicamente va a parar un archivo de módulo. Entre las diferentes versiones de núcleos se han estado desarrollando también diferentes formas de organizar y repartir los archivos de núcleos.

Las versiones 2.0 del núcleo han sido orientados a computadoras con procesadores intel (i386) y contemplando solamente hardware "estándard", más que todo controladores de discos (dispositivos bloque), de terminales (dispositivos de caracteres), dispositivos de red y algunos mas. La gran variedad de hardware, por ejemplo dispositivos de sonido ha requerido separar estos en un subdirectorio aparte en el núcleo 2.2.

A partir del núcleo 2.4 hubo una reestructuración fundamental, que toma en cuenta que el núcleo Linux ya está disponible para muchos diferentes arquitecturas de computadoras (diferentes procesadores), y separa elementos independientes de la arquitectura de elementos específico para el CPU o para los buses específicos para un cierto hardware en diferentes directorios.

También el paquete modutils depende de esta distribución y versiones anteriores de este software no encuentran los módulos de núcleos mas recientes.

### 17.3.3. Otros componentes

En el directorio /boot también se copian los archivos config-<versión>del núcleo, que contiene todos los parametros de compilación del núcleo en cuestión, y el archvio System.map-<versión>que contiene los símbolos exportados.

Los felices que utilizan el sistema Debian pueden contar con que la utilidad "kernel-package" copie estos archivos y los módulos automáticamente en el lugar correcto. "make-kpkg kernel-image" crea un paquete debian (.deb) en el directorio /usr/src, que puede ser "instalado" con el comando "dpkg -i kernel-image-<versión>_Custom.1.00_i386.deb". Este paquete contiene todos los componentes arriba mencionados, los copia en sus lugares correctos y ejecuta lilo para instalar el núclo en el sector de arranque. Si se trata de instalar un núcleo con una versión ya instalado, el programa se rehusa a copiar los módulos y pide reconfirmación. Si se está seguro de querer sobreescribir la versión actual del núcleo de esta versión no se debe insistir, sino primero borrar el directorio con los módulos por completo: rm -rf /lib/modules/<version>, de otra manera se mantienen posiblemente archivos de módulos adicionales en este directorio de los cuales depmod genera mensajes de error de dependencia a la hora de iniciar la computadora.

No es recomendable borrar los módulos del núcleo de instalación. "kernel-package" permite especificar una "sub-versión" para el núcleo a la hora de compilación, con lo cual el nuevo núcleo se distingue aunque sea de la misma versión que un núcleo actual En este caso se instalan los módulos en un directorio diferente. Esto se realiza mediante la opción:

```
make-kpkg --add-to-version <etiqueta> kernel-image
```

donde <etiqueta>puede ser escogido por el administrador del sistema, p.ej. "-actual", o "-1", "-2", etc.

El paquete "kernellab" permite la administración y regeneración automática de muchos versiones y configuraciones de núcleos en una sola computadora, por ejemplo para una máquina "servidor" de núcleos para una red grande y con diferentes computadoras.

## 17.4. Unidad Zip en puerto paralelo

## 17.5. SCSI y escaneadores

## 17.6. Puertos en serie, modems y Plug and Play

## 17.7. Tarjetas de sonido

## 17.8. Adaptadores de Red

## 17.9. Niveles de ejecución y demonios

[rol de demonios en el sistema. Funcionamiento detallado de init y uso de diferentes runlevel - niveles de ejecución.]

# 18. Programas

[opcional]

## 18.1. StarOffice

[importante]
[Libre equivalente a MS-Office, Instalación correcta]

## 18.2. Gnucash

[Programa de contabilidad, opcional]

## 18.3. Correo Electrónico

[Medio importante]
[Configuración de correo en: varias cuentas, para varios MUA, desde varios servidores externos via Pop3/Multipop, multidrop fetchmail, fetchmail individual por cuenta]

## 18.4. Fax, Voice

[Fax importante]
[Efax - sencillo, una computadora; hylafax - server, una solución para redes; Voice: no me ha interesado hasta ahora, posiblemente GNUe-telephony=Bayonne]

# 19. Bonboncitos

Este capítulo da información básica sobre algunos aspectos interesantes e inclusive importantes en el uso de computadoras en general y de Linux en especial. Sin embargo se trata de temas muy amplios o también arbitrariamente seleccionados porque son del interés del autor.

## 19.1. Respaldos

[¡Muy importante, ya que hay poco conocimiento práctico, poca práctica, y MUCHA necesidad!]

## 19.2. Fips

[Compagína con Instalación del Sistema Operativo, ya que permite instalar Linux posterior a una instalación existente de Windows sin necesidad de reinstalar Wi...]

## 19.3. TEX/LATEX, dvi y LYX

[Lo mejor en Procesamiento de Palabras, ¡really!, este documento ha sido elaborado con eso y en una velocidad y calidad incomparable con Wor... no StarWrite]

## 19.4. The Gimp

[El sustituto para Photoshop]

## 19.5. Linuxdoc

[Mhmmm... Cómo sacarle mejor provecho a un texto. Generar de una sola fuente una multitud de formatos y presentaciones: Web, TEX, rtf, ASCII, etc. Podría extender también sobre Literate Programming, y Estilo de documentación técnica en Informática (texinfo!!)]

## 19.6. ps-tools

[¿y también Display Postscript?]

# Parte IV.

# Anexos

# A. Glosario

**El byte** es la segunda unidad básica para medir volúmen de información, es la unidad más utilizada. La unidad básica es la mínima cantidad de información 0 o 1, es el Bit. 8 Bit se agrupan para formar un Byte (256 posibles entidades de información). 10 Bit expresan 1024 posibilidades, 1024 Byte son 1 KByte. 1K = 1024, 1k = 1000. El megabyte no tiene definición concisa, hay quienes la calculan como

1 M = KxK = 1024x1024 = 1048576

1 M = kxK = 1000x1024 = 1024000

La misma situación ocurre con Gigabyte, GByte.

**La unidad traditional de medida en medios para grabar archivos** es un bloque = 512 Byte = 1/2 KByte. "du" traditionalmente reporta bloques, en sistemas GNU reporta KBytes. Con la opción -b muestra bytes.

**\t** tabulador

**Chipset** Ciertas funciones de Hardware, como la generación de la señal video para el monitor, o el control de los buses de expansión ISA y PCI y el controlador de disco duro que tradicionalmente se realizaron mediante circuitos electronicos individuales son integrados en unos pocos circuitos de alta densidad de integración por algunos productores de componentes electronicos. Estos muchas veces no permiten el uso de componentes diversos, sino requieren que todos los componentes sean de la misma fábrica. Estos "juegos" de Chips (= sinónimo para circuito integrado) son los Chipset, que se denominan según el circuito principal y/o la fábrica. La alta complejidad a veces lleva a diseños que exponen fallas, deficiencias, o incompatibilidades.

**Hash de dispersión** Una secuencia de caracteres (bytes) puede ser interpretado como un número con muuuchas cifras. A través de operaciones matemáticas (divisiones con polinomios residuales y otros monstruosidades) se crea un número (el resto de la división) con una cantidad fija de cifras = cantidad fija de bytes: el hash, a veces también se le llama checksum. El hash debe ser diferente para cada secuencia de caracteres específica, quiere decir, si se cambia una "cifra" en la secuencia, también debe de cambiar el checksum. Una función de dispersión buena solo tiene muy "pocas" secuencias que producen el mismo checksum respectivo. El checksum entonces simboliza la secuencia de caracteres, es como una firma, o un código corto para ella. La secuencia de caracteres puede ser una clave, un paquete de datos en una transmisión de red, un archivo, etc.

**expresiones regulares** protector del carácter siguiente

## Comandos de ratón

**Click** apretar rápidamente un botón

**Doble-Click** apretar dos veces consecutivamente un botón

**Jalar** apretar – mover – soltar

**Drop** soltar un botón apretado

**Left-Click, Right-Click** apretar el botón izquierdo o derecho del ratón respectivamente.

**POSIX** el nombre "Unix" es el término más popular para designar una serie de sistemas operativos con una funcionalidad compatible. "Unix" es una marca registrada y no puede usarse como término para referirse a estos sistemas en general. Por otro lado existe un esfuerzo de estándardizacion de estos sistemas operativos llamado POSIX, que describe las funciones requeridas para que un sistema sea compatible con ... POSIX, pués. en la literatura a veces se utiliza "*i*x" o un comodín similar para estos sistemas (IRIX, Linux, HP-UX, Unix,...). El plural de Unix es "Unices".

**root** Hay algo de confusión sobre esta palabra. La usamos en dos contextos en los sistemas GNU:

1. La raíz de un arbol de directorios
2. la cuenta superuser, o cuenta administrativa

En el primer caso, es un término técnico que puede ser traducido como raíz de un arbol de directirios. El segundo caso, sin embargo es un nombre propio y no puede ser traducido. Es el nombre, que tradicionalmente se le asigna a una cuenta de usuario, que tiene el númerio id(entificador) de usuario 0, y el número id de grupo 0. Muchos programas suponen que esta cuenta se nombrada literalmente "root", y no puede ser traducida.

# B. Referencia rápida Emacs

## GNU Emacs Reference Card

(for version 20)

### Starting Emacs

To enter GNU Emacs 20, just type its name: emacs

To read in a file to edit, see Files, below.

### Leaving Emacs

| | |
|---|---|
| suspend Emacs (or iconify it under X) | C-z |
| exit Emacs permanently | C-x C-c |

### Files

| | |
|---|---|
| **read** a file into Emacs | C-x C-f |
| **save** a file back to disk | C-x C-s |
| save **all** files | C-x s |
| **insert** contents of another file into this buffer | C-x i |
| replace this file with the file you really want | C-x C-v |
| write buffer to a specified file | C-x C-w |
| version control checkin/checkout | C-x C-q |

### Getting Help

The help system is simple. Type C-h (or F1) and follow the directions. If you are a first-time user, type C-h t for a **tutorial**.

| | |
|---|---|
| remove help window | C-x 1 |
| scroll help window | C-M-v |
| apropos: show commands matching a string | C-h a |
| show the function a key runs | C-h c |
| describe a function | C-h f |
| get mode-specific information | C-h m |

### Error Recovery

| | |
|---|---|
| **abort** partially typed or executing command | C-g |
| **recover** a file lost by a system crash | M-x recover-file |
| **undo** an unwanted change | C-x u or C-_ |
| restore a buffer to its original contents | M-x revert-buffer |
| redraw garbaged screen | C-l |

### Incremental Search

| | |
|---|---|
| search forward | C-s |
| search backward | C-r |
| regular expression search | C-M-s |
| reverse regular expression search | C-M-r |
| select previous search string | M-p |
| select next later search string | M-n |
| exit incremental search | RET |
| undo effect of last character | DEL |
| abort current search | C-g |

Use C-s or C-r again to repeat the search in either direction. If Emacs is still searching, C-g cancels only the part not done.



1

## Multiple Windows

When two commands are shown, the second is for "other frame."

| | | |
|---|---|---|
| delete all other windows | C-x 1 | |
| split window, above and below | C-x 2 | C-x 5 2 |
| delete this window | C-x 0 | C-x 5 0 |
| split window, side by side | C-x 3 | |
| scroll other window | C-M-v | |
| switch cursor to another window | C-x o | C-x 5 o |
| select buffer in other window | C-x 4 b | C-x 5 b |
| display buffer in other window | C-x 4 C-o | C-x 5 C-o |
| find file in other window | C-x 4 f | C-x 5 f |
| find file read-only in other window | C-x 4 r | C-x 5 r |
| run Dired in other window | C-x 4 d | C-x 5 d |
| find tag in other window | C-x 4 . | C-x 5 . |
| grow window taller | C-x ^ | |
| shrink window narrower | C-x { | |
| grow window wider | C-x } | |

## Formatting

| | |
|---|---|
| indent current line (mode-dependent) | TAB |
| indent region (mode-dependent) | C-M-\ |
| indent sexp (mode-dependent) | C-M-q |
| indent region rigidly *arg* columns | C-x TAB |
| insert newline after point | C-o |
| move rest of line vertically down | C-M-o |
| delete blank lines around point | C-x C-o |
| join line with previous (with arg, next) | M-^ |
| delete all white space around point | M-\ |
| put exactly one space at point | M-SPC |
| fill paragraph | M-q |
| set fill column | C-x f |
| set prefix each line starts with | C-x . |
| set face | M-g |

## Case Change

| | |
|---|---|
| uppercase word | M-u |
| lowercase word | M-l |
| capitalize word | M-c |
| uppercase region | C-x C-u |
| lowercase region | C-x C-l |

## The Minibuffer

The following keys are defined in the minibuffer.

| | |
|---|---|
| complete as much as possible | TAB |
| complete up to one word | SPC |
| complete and execute | RET |
| show possible completions | ? |
| fetch previous minibuffer input | M-p |
| fetch later minibuffer input or default | M-n |
| regexp search backward through history | M-r |
| regexp search forward through history | M-s |
| abort command | C-g |

Type C-x ESC ESC to edit and repeat the last command that used the minibuffer. Type F10 to activate the menu bar using the minibuffer.

# C. Referencia rápida vi

Este archivo es un resumen y una traducción del documento /usr/doc/nvi/vi.beginner

[next] [prev] [prev-tail] [tail] [up]

# Apéndice C
# Referencia rápida vi

Este archivo es un resumen y una traducción del documento /usr/doc/nvi/vi.beginner

| Acción | Tecla |
|---|---|
| entrar en vi desde el shell | vi filename |
| salir y guardar | ZZ o :wq<Enter> |
| salir y descartar | :q!<Enter> |
| salir (no hubo cambios) | :q^M |
| guardar sin salir | :w^M |
| guardar con otro nombre | :w nombre^M |
| **mover el cursor:** | |
| derecha | l |
| izquierda | h o C-H |
| abajo | j o C-N |
| arriba | k o C-P |
| **insertar texto:** | |
| ante cursor | i |
| comienzo de la línea | |
| después del cursor (añadir) | a |
| al final de la línea | |
| después de la línea actual | o |
| ante la línea actual | O |
| **borrar el caracter:** | |
| debajo del cursor | x |
| a la izquierda del cursor | X |
| borrar n caracteres | #x o #X |
| **movimiento ámplio:** | |
| desplazar línea superior de la pantalla | ^Y |
| desplazar línea inferior de la pantalla | ^E |
| subir media pantalla | ^U |
| bajar media pantalla | ^D |
| mover cursor a la línea superior | H |
| mover cursor a la línea inferior | L |
| mover cursor a la línea en medio | M |
| **marcas:** | |

| Acción | Tecla |
|---|---|
| juntar dos líneas | J |
| **encontrar cadena en el archivo:** | |
| busquando adelante | / ...string... /^M |
| busquando hacia atrás | ? ...string... ?^M |
| repetir la última búsqueda | n |
| repetir la última búsqueda pero en dirección opuesta | N |
| **buscar caracter en línea** | |
| busquando adelante | fc |
| busquando hacia atrás | Fc |
| repetir la última búsqueda de caracter | |
| **sustituciones** | |
| sustituir un caracter con el caracter x | |
| sustituir un caracter con un texto | s ...text... |
| sustituir n caracteres con un texto | |
| sustituir caracteres uno por uno con texto | R ...text... |
| **deshacer** | |
| deshacer todos cambios en la línea actual | U |
| deshacer el último cambio | |
| **movimiento ámplio :** | |
| adelantar una "pantalla" | ^F |
| retroceder una "pantalla" | ^B |
| adelantar una línea | |
| retroceder una línea | - |
| al inicio de la línea | 0 |
| al final de la línea | $ |
| adelantar una palabra | w |
| adelantar una palabra, ignorando puntuación | W |

| | | | |
|---|---|---|---|
| marcar la posición actual, nombrarla x | mx | retroceder al final de la palabra anterior | e |
| mover cursar a la línea marcada x | ☐ x | retroceder palabra ignorando puntuacioin | E |
| mover cursor al caracter marcado con x | ☐ x | retroceder una palabra | b |
| mover al comienzo del archivo | 1G | retroceder una palabra ignorando puntuación | B |
| mover al final del archivo | G | regresar a la última línea modificada | ☐ |
| mover a la línea 23 del archivo | 23G | | |

redibujar pantalla:

| | |
|---|---|
| centrar la línea del cursor | z^M |
| superior de la pantalla | z. |
| en medio de la pantalla | \- |
| inferior de la pantalla | z- |

| Caracteres especiales en cadenas de búsqueda: tienen que ser escapado con \ para usarlos directamente | Caracter |
|---|---|
| comienzo de la línea | ^ |
| final de la línea | $ - (símbolo dolar) |
| cualquier caracter | . - (punto) |
| el caracter de escape mismo | \ - (fleca) |
| para encontrar cadenas según comodínes | [ |
| ditto | ] |
| ditto | * - (asterisco) |

[next] [prev] [prev-tail] [front] [up]

Caracteres especiales en cadenas de bu'squeda:tienen que ser escapado con \ para usarlos directamente Caracter

comienzo de la li'nea ^final de la li'nea $ - (si'mbolo dolar) cualquier caracter . - (punto)el caracter de escape mismo \ - (fleca) para encontrar cadenas segu'n comodi'nes [ditto ] ditto * - (asterisco)

# D. Estándard de jerarquía de sistema de archivo

```
/                       directorio raíz
 `--bin                 comandos (binarios) esenciales de (todos) usuarios
 `--boot                archivos estáticos del boot loader
 `--dev                 archivos de dispositivos
 `--etc                 configuración específica del host
 |   |
 |   `--X11             configuración para el sistema X-Windows
 |   `--opt             configuración para /opt
 `--home                directorios personales de usuarios (opcional)
 `--lib                 bibliotecas compartidas esenciales
 |   |
 |   `--modules         módulos cargables del núcleo
 `--mnt                 punto de montaje temporal
 `--opt                 paquetes software adicionales
 `--proc                sistema virtual para información de núcleo y procesos
 `--root                directorio "home" para super-usuario
 `--sbin                binarios de sistema esenciales (para el arranque)
 `--tmp                 archivos temporales
 `--usr
 |   |
 |   `--X11R6           X-Windows, Versión 11, Revisión 6
 |   |   |
 |   |   `--bin         binarios de X-Windows
 |   |   `--lib/X11     bibliotecas específicas de X-Windows
 |   |   `--include/X11 archivos de inclusión de X-Windows
 |   `--X386            X-Windows, Ver. 11, Rev. 5 plataformas x86
 |   `--bin             la mayoría de los comandos de usuario
 |   |   |
 |   |   `--mh          commandos para el sistema de correo MH
 |   |   `--X11         enlace simbólico a /usr/X11R6/bin
 |   `--games
 |   `--include         archivos de inclusión estándard
 |   |   |
 |   |   `--X11         enlaces simbólico a /usr/X11R6/include/X11
 |   |   `--bsd         archivos de compatibilidad con BSD (en caso necesario)
 |   |   `--g++         archivos de inclusión de GNU C++
 |   `--lib             bibliotecas para programación y paquetes
 |   `--local
 |   |   |
 |   |   `--bin         binarios locales
 |   |   `--games       binarios de juegos locales
 |   |   `--include     archivos de inclusión locales
 |   |   `--lib         bibliotecas locales
 |   |   `--sbin        binarios de sistema locales
 |   |   `--share       jerarquía independiente de arquitectura local
 |   |   `--src         código fuente local
 |   `--sbin            binarios de sistema estándard no esenciales
 |   `--share           datos independientes de la arquitectura (plataforma)
 |   |   |
 |   |   `--dict        listas de palabras
 |   |   `--doc         documentación diversa
 |   |   `--games       datos estáticos para juegos (/usr/games)
 |   |   `--info        directorio primario para el sistema GNU Info
 |   |   `--locale      información de localización
 |   |   `--man         manuales (unix) en línea
 |   |   |   |
 |   |   |   `--man1    programas de usuario
 |   |   |   `--man2    routinas del sistema
 |   |   |   `--man3    routinas de bibliotecas
 |   |   |   `--man4    archivos especiales (de configuración etc.)
 |   |   |   `--man5    formatos de archivos
 |   |   |   `--man6    juegos
 |   |   |   `--man7    diversos
 |   |   |   `--man8    administración del sistema
 |   |   `--nls         soporte de lenguaje nativo
 |   |   `--misc        datos independientes de arquitectura diversos
 |   |   `--terminfo    base de datos de terminfo
```



```
|   |   `--tmac         macros troff no distribuidos con groff
|   |   `--zoneinfo     información y configuración de zonas de tiempo
|   `--src              código fuente
|       |
|       `--linux        código fuente del núcleo Linux
`--var                  datos variables
    |
    `--account          bitácoras de cuentas (opcional)
    `--cache            datos cache de aplicaciones
    |   |
    |   `--fonts        fuentes generados localmente
    |   `--man          páginas man formateados localmente (opcional)
    |   `--www          datos cache www o datos proxy
    |   `--<paquete>    datos cache de un paquete específico
    `--crash            vuelcos de fallas de sistema (opcional)
    `--games            datos variables de juegos
    `--lib              información de estado variable
    |   |
    |   `--<editor>     Estado y archivos respaldo de editores
    |   `--misc         datos diversos de estado
    |   `--xdm          datos variables de administrador de pantalla X
    |   `--<pkgtool>    archivos soporte de empaquetado
    |   `--<paquete>    datos estáticos para paquetes y subsistemas
    `--lock             archivos de enllavado
    `--log              archivos y directorios bitácoras
    `--mail             apartados de correo de usuarios
    `--opt              datos variables para /opt
    `--run              archivos variables de corrida de procesos
    `--spool            datos cola de aplicaciones
    |   |
    |   `--lpd          directorio cola para impresión
    |   |   |
    |   |   `<impresora> colas de impresión para impresora específica (opcional)
    |   `--mqueue       cola de correo saliente
    |   `--news         cola de News (Noticias)
    |   `--rwho         archivos rwhod
    |   `--smail        directorios cola para smail
    |   `--uucp         directorio cola para UUCP
    `--tmp              archivos temporales preservados durante re-inicios
    `--yp               base de datos para Network Information Service
```

# E. Visualizadores

Para la gran variedad de tipos y formatos de archivos hay una igual variedad de utilidades de visualización.

Unix no utiliza en primer lugar el concepto de "extensión" del nombre de archivo para distinguir los formatos, sino deduce del contenido del archivo su tipo. Sin embargo se utiliza las extensiones para facilitar a la usuaria humana el reconocimiento y la administración de sus archivos.

La utilidad "`file`" cuenta con una base de datos de "números mágicos" (magic number), que son los primeros tantos caracteres de un archivo y que generalmente se distinguen según el tipo del archivo.

`file` *nombre*

Utiliza esta base de datos para escribir en la salida estándar el tipo del archivo.

Otro mecanismo que está ganando importancia es la norma MIME (Multipurpose Internet Mail Extensions), que prove una nomeclatura estandardizada para los diferentes formatos y permite una asociación entre tipos de archivos y aplicaciones para su manipulación y/o "visualización" respectiva. Esto último es importante, porque permite que en diferents computadoras, con sistemas operativos o capabilidades diferentes de software instalado siempre se utiliza el "visualizador" adecuado para un cierto tipo de archivos según la asociación MIME individual configurada en esta máquina.

Hay visualizadores especializados para solo un cierto tipo de archivos, pero también hay utilidades genéricas que se adaptan automáticamente a una gran variedad de diferentes formatos de archivos, muchas veces sirviendose de los visualizadores especializados. Los administradores de archivos (Filemanager) normalmente cuentan con un visualizador integrado.

El concepto de un visualizador (browser) es hacer visible de manera rápida el contenido de un archivo en la pantalla y eventualmente imprimirlo, pero no incluye la edición del contenido del archivo en cuestión.

El concepto de un paginador (Pager) es orientado a formatos textuales o flujos (streams) de caracteres y es, que el paginador divide el documento en partes que alcanzan verse en una pantalla o imprimirse en una hoja de papel de un cierto tamaño respectivamente.

**more** es el paginador tradicional de unix. Su nombre se deriva de que presenta una página de texto y después una línea con el texto "more" = más. Si se aprieta cualquier tecla (menos "q") muestra una página "más". Con "q" interrumpe la paginación.

**cat** es un filtro unix para concatenar varios archivos, pero puede usarse para imprimire un texto (corto) en la pantalla

**head/tail**

**less** "more", pero con todas las sofisticaciones imaginables, en especial, puede navegarse el texto hacia adelante **y** hacia atrás.

**vi/emacs** y otros editores pueden usarse como visualizadores y tienen una opción de abrir un archivo de solo lectura. Emacs tiene un sinnúmero de filtros para formatear (y manipular) diferentes tipos de archivos no-textuales. En especial mencionamos el modo "hexl" que permite visualizar archivos (binarios) en forma hexadecimal, y el modo sgml, que permite visualizar archivos sgml, html, xml, etc.

**mc** es un administrador de archivos en modo ascii/gráfico. Su forma de uso es copiado de Norton Commander (mc = midnight comander) pero tiene funcionalidad muy superior a este (enlaces ftp, http, nfs, . . . ). mc prove visualizadores para varios tipos de archivos y puede configurarse para visualizadores arbitrarios. En un sistema linux decentemente preconfigurado utiliza una vasta gama de visualizadores individuales para casi toda clase de archivos.

**lynx** es un navegador html ascii, que utiliza mime para visualizar archivos arbitrarios. De esta manera pueden visualizare los archivos en modo texto (p.ej. convertir un texto MS-Word en forma textual), pero también pueden lanzarse aplicaciones gráficas si se utiliza lynx en ambiente X-Windows.

**mozilla** o netscape realizan la misma tarea en modo gráfico en X-Windows.

**gv,**

**gqview,-TheGimp** son visualizadores para archivos gráficos. eog = Eye of the Gnome, eeyes = Electric Eyes. Los últimos dos no solo son visualizadores sino programas potentes para generar y manipular imágenes/archivos gráficos, mientras los primeros tienen facilidades de manejo de cantidades de imágenes, como p. ej. la de crear listas de iconos de archivos en un directorio (thumbnails). Hay visualizadores de gráficos que pueden usar las capacidades gráficas de la tarjeta SVGA, desde una terminal virtual de Linux: zvg

**man, info**

**tkman, ghelp** ameno para usar. ghelp es el visualizador de ayuda del Gnome Desktop y puede visualizar archivos man, info, y html.

# F. Introdución balazo en BNF

texto sencillo es obligatorio y no puede ser omitido

[texto opcional] está encerrado entre paréntesis cuadrados.

elemento... un elemento seguido por puntos suspensivos puede ser repetido las veces que sea deseado.

Ejemplo 1:

```
cp [OPCION] origen destino
cp [OPCION] origen directorio-destino
```

Significa: que hay dos formas de ejecutar el comando cp. la primera con precisamente dos parametros, la segunda con una cantidad arbitraria de orígenes. Pueden haber opciones, pero no necesariamente. Las opciones son un asunto aparte, declarado en un (ficticio) texto titulado OPCION, por eso están puestos en mayuscula. Tomar en cuenta, que entre comando, opción(es) y parámetro tiene que ponerse espacio en blanco.

Ejemplo 2:

```
bouncesaying error [program [arg ]]
```

El comando puede tomar cualquier de las siguientes formas:

```
bouncesaying error
bouncesaying error program
bouncesaying error program arg
```

Notese que en esta presentación la lista de argumentos (`arg...`) puede consistir en una mezcla de opciones y parámetros. La notación no lo especifíca al detalle.

# G. Introducción balazo en expresiones regulares

^ comienzo de la línea

$ final de la línea

. cualquier caracter

Ejemplos:

```
grep c archivo
```

imprime todas las líneas de *archivo* que contienen la letra 'c'.

```
grep ^.c archivo
```

imprime todas las líneas de *archivo* que tienen como segundo caractér la letra 'c'.

```
grep ' $' archivo
```

imprime todas las líneas de *archivo* que tienen por lo menos un espacio en blanco al final de la línea

* repite el caracter anterior cero o mas veces

+ repite el caracter anterior uno (1) o mas veces

{*n*, *m*} repite el caracter anterior mínimamente $n$ y máximamente $m$ veces

Ejemplos:

```
grep 'campos*' archivo
```

imprime todas las lineas de *archivo* que contienen la palabra campo y/o la palabra campos.

```
grep c+ archivo
```

imprime todas las líneas de *archivo* que contienen 'c', 'cc', 'ccc', etc.

```
grep -r -v -i -l "hola .*jorge" *
```

encuentra todos los archivos que contienen líneas como: 'hola jorge', 'hola querido jorge', 'hola estimado jorge', etc.

[*lista de caracteres*] aplica, si el caractér en la posición respectivo concuerda con cualquiera de los caracteres en la lista

`[:digit:]` lista de caracteres predefinida que contiene todos los digitos

`[:alnum:]` lista de caracteres predefinida que contiene todas las letras y digitos

`[:space:]` lista de caracteres predefinida que contiene todos los caractéres que producen espacio en blanco: tabuladores, avances de página, etc.

... hay varias listas predefinidas mas.

Ejemplos:

```
grep 'bols[oa]' archivo
```

imprime todas las líneas de *archivo* que contienen la palabra bolso y/o la palabra bolsa.

```
grep '^[:digit:]+.*[:space:]' archivo
```

imprime todas las lineas de *archivo* que comienzan con uno o mas digitos, seguidos probablemente por cualquier letra, seguido por al menos un espacio en blanco.

[*n*..*m*] lista de caracteres, comenzando con *n* y terminando con *m*. Ejemplo: `[0..9]` es equivalente a `[:digit:]`

Ejemplo:

```
grep '^[a..mA..M]' archivo
```

imprime todas las líneas de *archivo* que comienzan con una letra de la primera mitad del abecedario.